UNIVERSIDAD CARLOS III DE MADRID

Escuela Politécnica Superior

Final Degree Project

Assessing the Peering Evolution in the AFRINIC Region

Author: Mª Cristina Márquez Colás

Academic tutor: Albert Banchs Roca

IMDEA Networks tutor: Pierre Francois

Bachelor's degree in Telecommunication Technologies Engineering

Acknowledgements

Immeasurable appreciation and deepest gratitude for the help and support are extended to the following people who have contributed to my personal and professional growth along the years:

My parents, of course, for all their love, support and patience. Without you this would have never been possible.

My brother, who has helped me and advised me since I have memory.

PhD. Pierre Francois, who gave me the opportunity to join his team as well as for all the support, experience and knowledge. I take this opportunity to express my sincere thanks to all the team who has embraced me as one of them from the very beginning.

PhD. Albert Banchs, for providing me with all the facilities to develop my research at IMDEA Networks Institute.

The professors: Jair Montoya, Franchesca Collet, Mei Ling, Judith Liu, Tobías Koch, Alejandro Lampérez, Daniel Díaz-Sánchez, Matilde Pilar Sánchez, Jesús Arias and Raquel Pérez Leal. You all illustrate what passion means for our job and each one of you transmit it in a unique way, with your own essence. I wish I would be able to be as good as you sharing it in the future.

My best friends, who understood perfectly how important and how much sacrifice this bachelor degree needed. Thank you for making me feel alive between study weekends and for accepting me as I am, contributing to create who I am today.

My old friends, for pushing me to go out in holidays at least one day.

My colleagues and classmates at the University. You added everyday a bit of humor when stress, deadlines and exams were present. I hope we can meet again after University to remember good times.

My high-school and school professors: Mari Luz, Nieves, Jesús Cantos, Felipe, Javier, Fermín, etc. Did you imagine me writing a hundred pages report in English? Probably not, but you all believed in me and discovered my potential. *You would be whoever you want to be*. *You are a machine*. I still remember those words which have helped me a lot along my whole life encouraging me .

I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their hand in this venture. I appreciate as well those who complicated my life, you made me realize I can deal with everything without any exception.

Abstract

The poor quality of Internet access appears as an obstacle in many African countries for their development. Many organizations aim at assisting African ISP providers and universities to solve this problem. So far, few previous studies have targeted the African continent. The purpose of this project is to complement those studies by learning from historical routing data the peering habits among local ISPs. By considering this routing data covering the last decade we compute diverse statistics and analyze the growth of involved African Internet Exchange Platforms.

Our results show that almost all the prefixes appear since their allocation date and most of them have appeared on 2015 as the year last of appearance. Moreover, the most frequent prefixes come from South Africa, Nigeria and Egypt. Also, the IXPs mostly used for peering (JINX and CINX according to our dataset) have more reallocated prefixes than the rest of the IXPs. In addition, the newest IXPs are TunIXP, SIxP, and DINX. Hence, they are still growing, whereas the IXPs who have been peering the earliest (JINX and KIXP) show a drop in the evolution.

Such findings are essential for taking suitable decisions aiming at empowering the Internet underlying structure knowledge in the region. They can easily help ISPs to choose at which IXP to peer next, or be used by the stakeholders to evaluate the growth of their IXP in comparison with others. Moreover, one can determine as regional IXPs those at which we discovered most prefixes and origin ASes connected to and boost them.

Key words: IXP, peering evolution, African Internet, PCH historical data.

General index

List of figures

List of tables

List of terms

API: Application Programming Interface.

AS: Autonomous System.

ASN: AS Number.

BGP: Border Gateway Protocol.

CC: Country Code.

DNS: Domain Name System.

EGP: External Gateway Protocol.

HTML: HyperText Markup Language.

IANA: Internet Assigned Numbers Authority.

IATA: International Air Transport Association.

IGP: Interior Gateway Protocol.

IP: Internet Protocol.

ISOC: Internet SOCiety.

ISP: Internet Service Provider.

IXP: Internet eXchange Point.

OS: Operating System.

PCH: Packet Clearing House.

RIR: Regional Internet Registry.

RIS: Routing Information Service.

URL: Uniform Resource Locator.

Chapter 1: Introduction

We discuss in this chapter the motivation of our study, which consists in analyzing the evolution of peering among local networks in the AFRINIC region. Next, we shed light on the objectives and the structure of this thesis.

1.1 Motivation

In the related work, different studies have been assessing the Internet [1] [2] [3]. While some have tried to map the whole Internet, others have targeted some specific regions such as the US [4]. Most of those works were based on traceroutes data.

Earlier mapping efforts were mainly twofold. Some attempts consisted in inferring the router-level map of the whole Internet from traceroutes performed either by a single monitoring node [5] or a set of nodes (vantage points, traceroute servers) deployed at arbitrary locations. In both cases, destinations addresses are extracted from external databases such as routing tables dumps or database web servers [3]. The second objective of such works was to minimize the number of measurements needed. It was achieved by analyzing the utility of adding vantage points and destinations to discover the Internet topology. The results of [4] have proved that focusing on a specific set of ISPs gives refined results.

So far, the common specificity of these projects is to have under-involved African countries. A few studies have indeed been targeting the Interdomain topology on the African continent [6] [7]. More recently, some researchers [7] have highlighted the remaining lack of interconnectivity among local African ISPs, as well as the use of existing Internet eXchange Points (IXPs) and the appearance of some recently established ones.

Our study complements these recent works as we analyze in detail the growth of African IXPs over the last decade. Packet Clearing House (PCH), a non-profit institute, deploys route-collectors at most of the IXPs all over the world and collects BGP routes exchanged by its peers at those IXPs. Since this dataset (termed PCH dataset in this thesis) is publicly available [8], we used it for our purposes.

It contains 'show BGP' of PCH boxes (route-collectors) deployed at diverse Internet Exchange Points (IXPs). Our main objective in this longitudinal study is to shed light on the view of the African Interdomain routing seen from each IXP or set of IXPs on the continent.

We first geolocate the different IXPs in the PCH dataset and classify them per Internet region (RIR – Regional Internet Registries). We then analyze and compute various statistics based on the BGP routes exchanged by peers and announced to PCH route-collectors at each IXP. Thanks to those results, we will be able to identify how local ASes have been peering over time and which IXPs are mostly used.

1.2 Objectives

We aim at computing diverse statistics based on historical routing data collected from IXPs in order to better understand how local ASes have been peering over time. Our results will definitively be helpful for some Institutions in order to make suitable decision and empower the Internet in the region. For instance, they could easily elect as regional IXPs those where we found most local networks connected to.

As an additional objective, we will track the growth or the death of existing exchange points. The authors of the paper "*On the Diversity of Interdomain Routing in Africa*" [7], have discovered many African IXPs. We can also confirm whether those IXPs were existing before given our analysis. To achieve our goals, we compute the following statistics using data sources covering the last 10 years:

- Time difference between prefix allocation and Appearance on the Internet.
- Time difference between prefix Allocation and Appearance in the data collected by PCH route-collectors deployed at an African IXP.
- Number and list of different prefixes visible in the data collected by PCH route-collectors deployed at an African IXP.
- Number and list of Autonomous Systems (ASes) visible in the data collected by PCH route-collectors deployed at an African IXP.
- Prefix and Autonomous System Numbers (ASNs) growth statistics per year at IXPs in PCH dataset.
- Unique number of prefixes and ASNs that appear at an IXP in consecutive and non-consecutive years from 2005 to 2015 in PCH dataset.
- Ratio of African ASNs assigned to the country that are visible at an IXP in PCH dataset.
- Number and ratio of ASNs by country assignment (local vs. external).

For each result, we will explain in detail both the process to obtain them and the information that can be learnt or extracted from them at the 5th and 6th chapters.

1.3 Structure of the thesis

The thesis structure is broken down in 7 chapters in order to facilitate the reading comprehension and to identify the distinct parts required for accomplishing the project objectives. They are slightly described below:

- Chapter 1: Introduction. It describes needs, motivations and goals of the project as well as the document structure.

- Chapter 2: Problem approach. It defines the repository where the data is downloaded and the reasons behind that decision. It also describes the resources used for achieving our objectives.
- Chapter 3: Data collection and classification methodology. It defines the process for downloading and classifying the dataset.
- Chapter 4: Data parsing and storage. It describes the parsing process and the database structure. Additionally, we describe the RIRs database structure required.
- Chapter 5. Statistics (part I). In this chapter the first four objectives are accomplished. We describe the whole process for reaching each one of these objectives that can be accomplished independently of the remaining statistics.
- Chapter 6. Statistics (part II). It describes in detail how the remaining objectives were reached. Some of these statistics required the results given in the previous chapter.
- Chapter 7: Conclusions. In this chapter, we summarize the thesis content, the issues encountered, analyze the fulfillment of the objectives, and we define future lines of work.

Chapter 2: Problem approach

We discuss in this chapter the reasons behind the choice of PCH data as BGP feeds, the technical knowledge required for understanding the complete work and the tools used for the correct development of the thesis.

2.1 The choice of PCH dataset

In the literature, many works have been relying on RouteViews raw data [5]. In fact, RouteViews contains in MRT files format the BGP routes (RIBs – Route Information Databases and UPDATEs) exchanged at IXPs collected daily from 2004 to 2015.

In RouteViews BGP feeds from a total of 11 IXPs are collected. However, among those IXPs, only two are in Africa: KIXP – Kenya Internet Exchange (Kenya), JINX – Johannesburg Internet Exchange (South Africa). It could be used for our study as well. But working on such data would have restricted our results for the whole African continent in only two countries.

In contrast, our research has led us to find a more complete database regarding the African continent. Since 2003, PCH has been peering at 159 IXPs covering 52 countries in 5 regions (AFRINIC, APNIC, LACNIC, ARIN and RIPE NCC). The data is a set of 'show ip bgp' published under a GZIP format. Only 3 (NWAX, TELXATL, DIXIE) of the IXPs in RouteViews are not involved. As for the African continent, 8 countries and 11 IXPs including KIXP and JINX are also involved in this dataset (table 1).

CC	Country	Cities	IXPs
ZA	South Africa	Cape Town, Johannesburg, Durban	CINX, JINX, DINX,
KE	Kenya	Nairobi	KIXP
MZ	Mozambique	Maputo	MIX
EG	Egypt	Cairo	CAIX
MW	Malawi	Lilongwe	MIXP
SD	Sudan	Khartoum	SIxP
TN	Tunisia	Tunis	TunIXP
NG	Nigeria	Ibadan, Lagos	IBIXP, NIXP

Table 1: African IXPs involved in PCH dataset.

Moreover, PCH is planning to deploy new boxes at 3 IXPs: Gambia, Tanzania and Rwanda. It is clear that using PCH dataset gives more completeness to our study. Note that the methodology (which will be explained at chapter 3) and all our scripts developed are compatible with the treatment of RouteViews data as well.

2.2 Definition of resources and previous knowledge needed

Once we defined the source of our dataset, we need to present the resources and previous knowledge needed for this thesis. It is first required to analyze the technical background, in order to understand the parameters studied for developing our analysis, as well as the tools used for storing and managing the data. Note that the Operating System (OS) could be replaced by some other OS and among others, the prerequisites of this project in terms of programming languages were Python, MATLAB, MySQL and HTML.

2.2.1 Technical background

In this section we introduce the background needed for understanding the whole project. We present a summary of the required terminology and the Internet protocol under study.

2.2.1.1 Basic terminology

Concept	Description
Internet Service Provider (ISP)	It is a company which mainly provides Internet connection to their customers, including personal and business access to the Internet. The customers could be both, other ISPs or individuals [9] [10].
Peering	It is a voluntary interconnection of separate ISPs aiming to exchange traffic between the customers of each network [9].
Internet Protocol (IP)	Protocol by which data is sent between computers on the Internet. Each computer (also known as host) has at least one IP address that identifies them uniquely on the Internet [11].
Autonomous System (AS)	IP network or group of IP networks possessing its/their own independent route policy [9].
Border Gateway Protocol (BGP)	Protocol that allows the exchange of routing information between ASes [9].
Internet eXchange Point (IXP)	Physical network access point through which ISPs connect their networks and exchange traffic. This structure minimize the ISPs traffic which should be delivered to their transit provider, and also minimize the average per-bit delivery cost [12].
Route-collector	Also referred as Public Route Server, systems that are publicly accessible, often via Telnet. It is able to also run pings, traceroutes, and "show ip bgp" commands [13].
Country Code (CC)	Two-letter suffix developed to represent countries and dependent areas, for use in data processing and communications [14]. Example: US (United States)
Regional Internet Registry (RIR)	Organization that manages the allocation and registration of Internet number resources (IP addresses and AS) within a particular region of the world [15]. The RIR system evolved over time, eventually dividing the world into five RIRs (see figure 1).

Table 2: Terminology involved in the document.

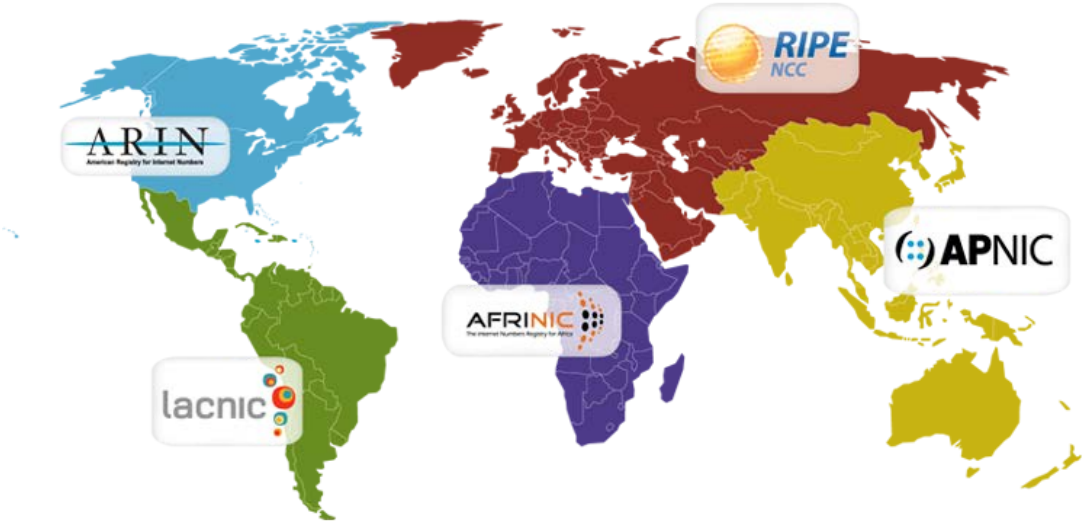


Figure 1: Regional Internet Registry (RIR) distribution [16].

2.2.1.2 BGP

The Border Gateway Protocol (BGP) is a gateway protocol designed for exchanging routing and reachability data among Autonomous Systems (AS) on the Internet. It may be used not only for routing between ASes (referred to as External BGP or eBGP), but also within an AS (referred to as Internal BGP or iBGP).

It decides the best path for a packet towards the destination based on the exchanged routing information between BGP enabled networking devices by means of a path-vector routing algorithm. Besides, it makes routing decisions taking into account the subnet reachability information, network policies, or rules configured by a network administrator [9, 17]

Based on the routing information (which includes the current route prefix for a destination as well as the AS path, and extra path attributes), each BGP speaker determines a reachable path while detecting and avoiding paths with routing loops [18].

Besides, BGP selects a single path as the best path to a destination network by default. Some of the extra attributes on the routing data are used in BGP best-path analysis. By altering them and configuring BGP policies the path selection can be influenced [19].

Among the information collected by BGP, the most important parameters in this document are described below according to the priority when selecting a route:

Parameter	Description
Local preference	Value assigned to a route by the AS's network administrator. The routes with the highest local preference values are selected [9].
AS-path	Sequence of followed ASes to reach a destination from a given IP address source. From the remaining routes which have the same local preference, the route with shortest AS-path is selected [9].
Origin AS	It is the AS where the route is originated. It is normally placed on the right-most side of the AS path [20].
Next AS	It is the adjacent AS to which we have to send the information in order to follow the AS-path. It is normally placed on the left-most side of the AS path [20].
Origin	It is the parameter which refers to the protocol that determined the route at an AS [9]. There are three possible values: IGP (Interior Gateway Protocol), EGP (External Gateway Protocol) or incomplete.
Next-hop router	From the remaining routes (all with the same local preference and the same AS-path length), the route with the closest next-hop router is selected [9].
IPv4 address	Route IP address number for a destination [21].

Table 3: Description of relevant BGP parameters.

2.2.2 Software tools

In this section we describe the software tools used and their configuration when needed (their cost is detailed in an annex section). Note that most of the tools are open source.

2.2.2.1 Python

According to its webpage, "Python is an interpreted, object-oriented, high-level programming language with dynamic semantics". It is generally used for scripting or as a language to connect existing components together [22], which is exactly what we desire.

The main characteristics that make this language attractive are:

- Its syntax facilitates readability and therefore decreases the cost of program maintenance.
- The variety of modules and packages supported encourage program modularity and permit code reuse.
- The absence of a compilation step.
- The debugger is written in Python itself and debugging Python programs is easy i.e. a bug or an incorrect input will never cause a segmentation fault. Instead of that, the interpreter will raise an exception when it discovers an error. When the exception is not catch, the interpreter prints a stack trace.

Python is often compared to other languages such as Java, JavaScript, or C++. In this section there are brief comparisons to each of languages, concentrated on language issues only [23].

- Java. Java programs are generally expected to run faster than Python programs, although they also take much more time to develop. Java programs are typically 3-5 times larger than equivalent Python programs. This difference can be attributed to Python's built-in high-level data types and its dynamic typing.

- JavaScript. As Python, JavaScript supports a programming style that uses simple functions and variables. However, JavaScript supports writing shorter programs and worse code reuse, whereas Python via an object-oriented programming style improves these features.

- C++. Almost everything commented for Java also applies for C++. Most importantly, while Java code is typically 3-5 times larger than equivalent Python code, C++ code is often 5-10 times larger than equivalent Python code.

Finally, we summarize the libraries needed for developing this thesis:

Library	Usage
os	This library allows to return the path where we are working in the server, create folders by means of a command-line argument [24], etc.
re	Library used for searching in the text some patterns and replace them [25].
MySQLdb	It is an interface that provides a connection to a MySQL database from Python [26].
time	By means of the sleep method this library allows to suspend the execution of a script given a number of seconds [27].
datetime	By means of the classes date and datetime [28] we were able to establish the dates in such format that allows to compute differences if needed directly in MySQL.
gzip	It is a command that combined with *open* method allows reading the information of ".gz" files without decompressing the packet.
ipaddr	This library is useful for managing IP addresses, both IPv4 and IPv6.
netaddr	This library is used for working with network address [29].
math	This library was used for returning the ceil of a number as a float [30].
csv	This library allows reading, writing and appending data to .csv files. It also determines the number of line that can be read.

Table 4: Summary of Python required libraries.

Depending on the Python version installed, some of these packages are not present. The process to install one of them (*netaddr*) is explained here, as an illustration. We first download the package from the webpage [29], we extract the files with the required command ("*tar –zxvf <package-name>*") [31], we access the unzipped folder and as root we install it with the command "python setup.py install" from the console.

2.2.2.2 Databases: MySQL

Databases are organized collections of data whose contents can be easily accessed, managed, and updated [32]. A database is composed of one or more tables, which store the information according to a defined set of parameters in columns and rows. Columns are also referred as fields, and rows as registers. Fields accept different data types [33] of information, such as: booleans, indexes, integers, alphanumerics, dates, strings, among others.

The main characteristics of databases [34] are:

- Logical and physical data independence.
- Minimun redundancy.
- Concurrent access by multiple users.
- Data integrity.
- Complex queries optimized.
- Access security.
- Access via various programming languages.

There are some typical databases [35] described below:

Database type	Description
MySQL	Database characterized by its quick access to the information. It is a SQL database. It is also multi-thread and can support many queries in parallel.
Access	Database developed by Microsoft.
PostgreSQL & Oracle	Database useful for managing huge amounts of data. They are commonly used at intranets or big systems.
Microsoft SQL Server	Database created also by Microsoft, but more powerful than Access.
MongoDB	It is an open-source document database, and the leading NoSQL database [36].

Table 5: Databases type description.

We decided to use MySQL because of its velocity for accessing the information and the data types it allows to store. In fact, MySQL is used by CDN (Content Delivery Networks) such as Google or YouTube. Besides, the possibility to manage the database through Python given that it is open-source, its ACID (Atomicity, Consistency, Isolation and Durability) characteristics also encouraged us to select this database.

Our algorithms use the parsed data which has been stored in databases. The format structure and the tables' size matters since the resources were extremely limited at the beginning (cf. section 2.2.3).

2.2.2.3 MATLAB

MATLAB [37] is a high-level computing language and an interactive environment for algorithm development, which allows to explore and visualize information, perform simulations, analyze data, manipulate matrices, create user interfaces and communicate with hardware and with programs written in other languages (including C, C++, Java, Fortran and Python [38]). Besides, lately its capability has increased for creating VHDL (Very High Description Language) code or directly programming digital signal processors.

Regarding the tradeoffs [39] involved when working with MATLAB, we first have to consider the time consumption it requires. A programmer has to decide how to prioritize: either programming fast scripts which are efficient on resources, or invest less effort programming while waiting for long-term results.

Secondly, we need to take into account the use of memory. Indeed, when working with big amounts of data, we need to keep all the information available in memory at once. If the scripts are large, the amounts of data may be gigantic.

Therefore, contrary to the run-time, the tradeoffs in terms of developing time, conjointly with the use of resources by MATLAB are really important and it is a must to be aware of them. However, MATLAB is useful for data analysis, visualization, plotting and obtaining sensational results with a minimum effort.

In the project, the software is used mostly for plotting 2D graphs with the results stored in a file given a pre-defined data structure. Typically, the file contains two different types of information separated by commas (the first data corresponds to the x axis and the data after the comma corresponds to the y axis). Note that we could have used Octave (open source), but MATLAB software was a prerequisite of this project (as mentioned in annex A).

2.2.2.4 Operating System (OS): Ubuntu

Ubuntu is a Free Software Operating System that was established as an enterprise server platform in 2004. However, free software was not usual at most users' computers. This fact motivated Mark Shuttle worth to create a team of developers from one of the most established Linux projects and launched Ubuntu, which is a Debian-based Linux operating system [40].

Ubuntu has additionally special releases for servers, OpenStack clouds, televisions and mobile devices. All releases share common infrastructure and software, making Ubuntu an exceptional single platform that scales from consumer desktops to the cloud for business computing.

This OS was selected taking into account all the facilities provided because of the model of free and open-source software development and distribution. It is also important the facilities to recover documents erased, because of the automatic backups it creates after a document is changed or moved. Besides, there is useful free software to work with scripts in parallel such as Cron or Screen.

However, Ubuntu is scrutinized for not contributing enough to projects such as the Linux kernel or for privacy and security issues regarding the member agreement Canonical (sponsor of Ubuntu) entered into with Amazon where search results of the company appeared when users look up in their local drives [41].

2.2.2.5 GNU Screen

GNU Screen [42] is a software application that allows multiplexing several virtual terminals in a single console. Each virtual session can be accessed by means of a single terminal or a remote one.

Moreover, this tool is extremely useful when using a server in order to check the proper operation of a script running, since there is a scroll-back history buffer for each one of them. This scroll-back captures the outputs even if the window is not being watched. This tool is also recommended for accessing from various computers.

2.2.2.6 Cron

Cron is a job scheduler of processes in the background (i.e. as a system daemon) at given intervals. In order to execute the processes at a desired time; a specific hour, the repetition frequency and the path towards the script to be run should be given in the “crontab” file [43].

2.2.2.7 Google Charts

It is Google’s tool [44] for visualizing data. At the gallery, there is a considerable amount of free and simple charts. That is the reason why the most common use of this tool is to embed a JavaScript in a web.

In the project, it was used to draw 3D Pie Charts with the results obtained at item 6.6 modifying the HMTL code given at the webpage. In comparison with MATLAB, it was faster since we had just to replace the values directly in the HTML code.

2.2.3 Physical resources and configuration required

In this section we comment the mandatory physical resources with their characteristics (their cost is detailed in the budget annex section) and the configuration between the laptop and the server.

2.2.3.1 Physical resources

The characteristics of the servers and the laptop are broken down here in this section. The server characteristics are listed in table 6, whereas the laptop characteristics are listed in table 7.

Component	Characteristic
Processor model name	Intel® Xeon® CPU ES-2665 0 @ 2.40 GHz
Hard disk capacity	200 GB. An expansion to 4TB was needed since the data downloaded occupied around 1 TB and the backup space of the required databases is around 600 GB in total.
CPU velocity	2.40 GHz
RAM	25 GB
Cores	8
Operating System	Debian 7 (Wheezy), 64 bits (Ubuntu server)
Type of server	Dedicated. It basically contains the databases and the scripts needed for filling the database and developing the distinct statistics of this project.
Network Internet card	Gigabit Ethernet (1 Gbps)
Graphic card	VGA compatible controller: Advanced Micro Devices, Inc. [AMD/ATI] Cedar GL [Fireproof 2270] (prig-if 00 [VGA controller])

Table 6: Server characteristics.

Component	Characteristic
Processor model name	Intel® Core™ i7 CPU M 640 @ 2.80 GHz. It is a Quad Core.
Hard disk capacity	500 GB
CPU velocity	2.80 GHz
RAM	4 GB
Cores	4
Operating Systems	Ubuntu 14.04 & Windows 7 Professional, 64 bits
Network Internet card	1 Gbps
Graphic card	VGA compatible controller: NVIDIA Corporación GT218M [NVS 3100M] (red a2) (prig-if 00 [VGA controller])

Table 7: Laptop characteristics.

2.2.3.2 Configuration between physical resources

Once the physical resources are defined, it is required to establish the protocols needed for sending and receiving the information both at the servers and at the laptop via Internet connection. The protocols were:

- Internet Control Message Protocol (ICMP) [9]. It is a protocol generally used between network devices for sending error messages, such as identifying if a server is not available or whether a host is unreachable. In this project, this protocol was used for checking connectivity among devices.
- HyperText Transfer Protocol (HTTP) [9]. It is an application layer protocol used in each transaction of the World Wide Web by information systems.
- HyperText Transfer Protocol Secure (HTTPS). It is a protocol based in HTTP, designed for transferring hypertext data securely. HTTP and HTTPS protocols were used for downloading the dataset and classifying it as explained in chapter 3.
- Secure SHell (SSH). It allows a securely access and complete control of remote machines through the network [45]. In this project, this protocol was used for connecting securely the server from our laptop and controlling the cron jobs.

In order to connect the laptop to the server through a secure connection, we need to create a private and a public key. We did it with GitHub (following the steps recommended at the webpage [46]) despite it is not the main feature of this Web-based Git repository hosting service [47].

Finally, in order to edit, write and send information to the server from the laptop it is mandatory to give the appropriate permissions to the user. This task was accomplished with the *chown* and *chmod* commands in the terminal [48] [49].

Chapter 3: Data collection and classification methodology

This chapter details the steps followed for downloading the PCH dataset and the methodology for classifying the mentioned dataset.

3.1 Data collection download

Our first script was aiming at downloading the PCH dataset. In order to do so, we created a list which contained all the available years on the web page and also three dictionaries[1]:

- The first dictionary contains the PCH route-collector name (each route-collector has a unique name) as a key and the complete URL of a hidden document (where all the URLs of the content are placed) as a value.
- The second dictionary contains the file names of a route-collector (those names are under the format "<route-collector>.<date>.gz") as a key and their URL as a value. We need those two dictionaries in order to access each URL needed.
- The third dictionary contains a summary of the downloaded data at a given time. It has as a key the year at which the data was collected, and as a value we have a dictionary with the PCH boxes' names as keys and the list of route-collector file names as a value. This dictionary was created to constantly check whether the data was fully downloaded. Thanks to this dictionary we discovered that the webpage was being continuously modified (although not the data itself) when downloading the PCH dataset.

With this structure we developed an algorithm which for every year in the list, downloaded the document at PCH which contains the list of route-collectors (called *data.php?dir=%2FYYYY%2F*, where the Ys are replaced by a year number) at a desired location in our server, following the structure of the PCH webpage (see figure 2).

[1] Dictionaries, sets, lists and tuples are Python data structures.

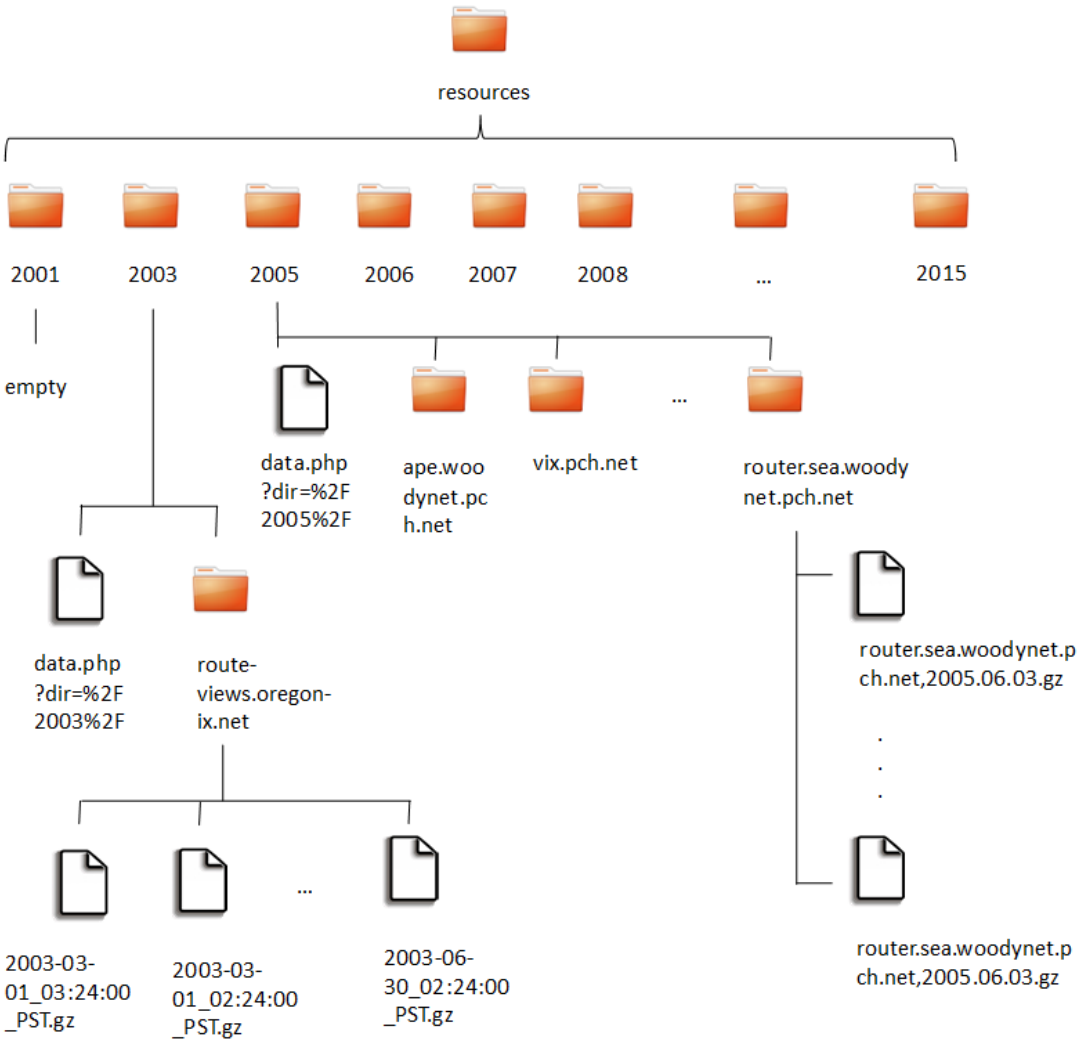


Figure 2: Visible PCH dataset structure [50] [51].

After that, we find that file and we open it for reading the distinct route-collectors names of a year. While reading them, we extract the URL for downloading their respective information and build the first dictionary.

Once we have created the first key of the first and third dictionary, we recreate the same PCH structure in our server. Therefore, we create a folder with the year as a name and we continue downloading the data of that year inside the folder given the links already collected by means of the command *wget* [52]. For each route-collector, there is a hidden file with the list of all the documents at that box. Then, we fill the second dictionary with the same process that filled the first one, and we download all the files of the desired box. Note that we wait a suspension time between the download of each file in order to avoid aggressing the server. We do this for every year until we complete the downloading and fill the dictionaries.

Since the URLs for 2003 were different, we developed another script for this special case. Besides, the file names and the data format collected for this year are completely different from all the other years. As it only contains data for one route-collector that is not located in the AFRINIC

region, we finally decided to remove it. Hence, we will focus in this thesis on the remaining years i.e. from 2005 to 2015 since no data was collected for 2004.

The advantage of creating the dictionaries given the document *data.php?dir=%2FXXXX%2* is that we only download the route-collectors data, and no more documents like the banner or the images at the webpage, optimizing the storage and the downloading of the dataset. We also reduced the downloading of not needed files extracting the URLs of the files at each route-collector. Although we could have created the same structure of the PCH webpage by means of the *wget* options, the best approach was the followed since we avoided downloading the non-desired files.

Furthermore, it was useful to download recursively the data by year in order to check if all data for each year was downloaded. If that was not the case, we could just download again the missing data thanks to the list where we store the year numbers. Due to the fact that the webpage was being continuously modified, we decided to download the data per year in parallel establishing the year we wanted to. This decision contributed to make our download script more efficient, since we are able to download a year dataset without having completed the download of another one.

3.2 Data classification

The downloaded dataset was not classified per region, so we geolocated each route-collector for developing our analysis. Our approach was to fill a table in the database called *locations*. That table contains for each route-collector their country code (CC).

3.2.1 Route-collectors geolocation

Geolocating a route-collector given its name at PCH is difficult since a unique pattern has not been defined. The name of a PCH route-collector in general is under the formats: “router.<site code>.woodynet.net” or “route-collector.<site code>.pch.net”. Note that <site code> is the nearest airport code in the first format, but it is not always the airport code in the second one. However, some <site codes> are a combination between city codes and IXP names.

It leads us to search for two letter CCs [53], city names [54], International Air Transport Association (IATA) codes [55] and IXP names [56] within the route-collectors names in order to geolocate them. Since this approach was not enough, we also searched for three letter CCs [57]. Next, we pinged the route-collectors for extracting their IP addresses and we tried to geolocate them using the Open IP Map, Maxmind, Team Cymru, and Whois databases and a reverse DNS method (detailed in table 10). Instead of pinging the route-collector, we could have done a reverse Domain Name System (DNS) lookup. Finally, we cross-checked the results with the ground truth deployment data obtained from PCH.

For geolocating them, we dealt with different document formats such as the ones presented in table 8.

Example number	Content of the document	Format in the document
1	European CCs	AL – Albania AD – Andorra
2	African CCs	Mozambique 508 MZ MOZ MZ Réunion 638 RE REU RE
3	US cities	Aberdeen, SD (ABR) Abilene, TX (ABI)
4	Airports	AYGA:GKA:GOROKA:GOROKA:PAPUA NEW GUINEA:06:04:54:S:145:23:30:E:5282 BGAM:N/A::ANGMAGSSALIK:GREENLAND:00:00:00:U:00:00:00:U:0000

Table 8: Document format examples of country and airport codes.

We had to be extremely cautious with the format of the documents and also with special characters such as 'ç', accents, tildes or '^'. For those cases, we decided to eliminate the accents and replace the character 'ç' by 'c' to avoid problems in the database. We also had to deal with extra issues at IXP list shown in table 9.

Example number	Case	Content structure: IXP name City Country
1	Three word names separated by '-'.	ACT-IX - Canberra Canberra Australia
2	Two word names separated by '-'.	ALB-IX Tirana Albania
3	Similar names.	AMS-IX Amsterdam Netherlands AMS-IX Kenya Mombasa Kenya
4	Separated words by spaces.	MEX - CEC Tokyo Japan ECIX Frankfurt Frankfurt Germany

Table 9: Issues regarding IXP names format.

All the possibilities in the IXP name had to be taken into account as well as the format name of each collector (normally separated by dots or dash '-') in order to search the patterns at each box's name.

In order to store the results of the classification, we needed to connect to the database with Python using a connector. With the connector to our database and a *cursor* object, we were able to create and insert all the results into the desired table [26]. The table that would store this first classification had the following structure:

Field	Type	Null	Key	Default	Extra
year	int(4)	NO		NULL	
location	varchar(255)	NO		NULL	
url	varchar(500)	NO	PRI	NULL	
country	varchar(40)	YES		NULL	
country_check	varchar(40)	YES		NULL	
ixp	varchar(20)	YES		NULL	
hint	varchar(40)	YES		NULL	
cc_code	varchar(2)	YES		NULL	

Figure 3: MySQL *year_<year_number>* table fields for classifying data according to the geolocation methodology.

Therefore this table (called *year_<year_number>*) will contain the year of the studied boxes, the collector's name in the field *location*, the path where the folder is at the server in the *url* field, the country found by the CC list in *country*, the IXP found on the route-collector's name in *ixp* given the IXP list and the airport code or the city name in the "hint" field. The columns called *country*, *country_check* and *cc_code* will have values in capital letters. The rest will be in the format given by the route-collector name or by the format defined as a key in the dictionary with the pattern found.

Due to the fact PCH was being continuously modified when downloading the data, we launched the classification in parallel. Thus, there will be a table per year studied. This was the best approach given the situation at PCH website.

In order to find the pattern to classify the boxes, we created a method called *classifier*. It received the dictionary with the collectors and their corresponding years, a dictionary with codes to apply and the name of the dictionary with codes as a string. With the first four codes we created four dictionaries that were saved in memory. The keys were the codes in capital letters and the values were lists of distinct possible formats of the country names, city names, airports names and IXP names respectively.

Regarding the dictionary given, the method follows a determined order, that's the reason why the name is given as a string. For example, if the dictionary containing the city names is given, we need to compare if the value saved in the hint column is found at the values in CC dictionary (where possible country names and cities of a country are listed). The reason why we check if it is in the values of the CC dictionary is that we want to have as a final result the country assigned to the collector. The same process is executed when dealing with the IATA or the IXP codes.

This method was modified to receive another dictionary because the previous one did not classify all the boxes. In fact, just 121 route-collectors out of 159 (76.1%) were classified with the four first lists. Thus, we applied the three letter CC [57] dictionary. The dictionary with these codes had the CCs in capital letters as keys and as a value a list with the possible formats of the countries appearance such as: uppercase format (since in the IATA code's document was possible), lowercase

format, spaces between the two-word country names, dots between each word, etc. This dictionary will need the two letter CCs dictionary or the airports one in order to fill later the "cc_code" field.

With this classifier, all the route-collectors were not classified, just 136 (85.53%). Hence, we pinged each route-collector for extracting the IP address and we tried to geolocate them using databases or methods listed in table 10. We traversed each database or method, whose input was an IP, till a valid code is found. The output of these databases was stored in the "country_check" field of year_<year_number> table. Note that these methods and databases were given by the research team. Hence, we cannot talk about their format, since we treat them as black boxes.

Order of used databases	Name of the databases or methods	Description
1	Open IP Map	It is a database based on crowd sourcing. Most of the participants are operators. They help geolocating routers by giving the corresponding CCs [58].
2	Reverse DNS	This method takes every router and looks for the name given an IP. If it is not found, the result on our database will be 'Unknown'.
3	Maxmind	Database used as a reference for geolocation [59].

4	Team Cymru	It is a database done by researches that have been working on regional reliable information for IP to AS mapping. It is strongly advised that CCs given by this database could not be the current IP locations. Thus, it will be necessary to cross-check the output [60]. The command required is:
		"Whois –h whois.cymru.com – v <IP_address>"
		
		Example of an output:
		`AS | CC | Registry | Allocated | AS Name`
		`27319 | US | arin | 2003-02-12 | ISC-F-AS`
		Figure 4: Whois output example.

5	Whois	This database [61] is based on the delegated files of the RIRs and it contains a list of every single domain currently registered in the world. The command required is: "Whois <IP_address>"

Table 10: Databases and methods used for geolocating PCH dataset.

However, three route-collectors did not return a CC with this method and three more had at "country_check" a CC different from the one saved at "country" (see figure 5). Therefore, six route-collectors (3.77%) were not geolocated with the current methodology.

location	country	country_check	ixp	hint	cc_code
200paul.woodynet.pch.net	NULL	US	NULL	NULL	US
amsix.woodynet.pch.net	US	US	amsix	NULL	NL
ape.woodynet.pch.net	NZ	US	ape	NULL	NZ
atlanta-ix.woodynet.pch.net	US	Unknown	NULL	atlanta	US
bdix.woodynet.pch.net	BD	US	bdix	NULL	BD
equinix-ashburn.woodynet.pch.net	NULL	US	NULL	NULL	US
equinix-la.woodynet.pch.net	NULL	US	NULL	NULL	US
equinix-sin.woodynet.pch.net	SG	US	NULL	sin	SG
hkix.woodynet.pch.net	HK	US	hkix	NULL	HK
jinx.woodynet.pch.net	ZA	US	jinx	NULL	ZA
laiix.woodynet.pch.net	US	US	laiix	NULL	US
linx.woodynet.pch.net	US	US	linx	NULL	GB
netnod.woodynet.pch.net	DK	US	netnod	NULL	SE
nota.woodynet.pch.net	ES	US	nota	NULL	US
npix.woodynet.pch.net	NP	US	npix	NULL	NP
nspixp2.woodynet.pch.net	NULL	US	NULL	NULL	JP
nyiix.woodynet.pch.net	US	US	nyiix	NULL	US
optiglobe.woodynet.pch.net	NULL	US	NULL	NULL	US
paix-ny.woodynet.pch.net	US	US	NULL	ny	US
paix.woodynet.pch.net	US	US	paix	NULL	US

Figure 5: MySQL year_<year_number> example table.

Hence, we cross-checked ground truth data of the location of each deployed collector got from PCH. It was necessary to check the ground truth data carefully since some of the route-collectors' information had two different CCs associated (e.g. router.trn.woodynet.net) and therefore they could be incorrectly classified later. Let's see an example:

Consorzio Top-IX (Italy)
Site Code: trn Country Code: FI
router.trn.woodynet.net
route-collector.trn.pch.net

Note that given the format "router.<site code>.woodynet.net", *trn* is the nearest airport code, which help us to figure out the city name. But, in the other notation, is not always the airport code. In this case we can say that the CC was not correct, but it was in RIPE region. The correct CC for this route-collector is IT.

We updated the database manually for those cases. We found that three route-collectors (1.885%) remained with the country we found at "country" column (due to the final geolocation method) and the other 1.885% route-collectors were updated according to PCH feedback. In conclusion, the final methodology script geolocated correctly 98.11% of the route-collectors before cross-checking it with PCH data, which it is a great performance.

3.2.2 Result of the geolocation methodology

Once the geolocation methodology is applied, the tables created are used to build a new one called "locations". It will match to every route-collector a unique CC corresponding to the country in which it is deployed or geolocated. This information will have associated a unique identifier.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
location	varchar(255)	NO	PRI		
cc_code	varchar(2)	NO	PRI		

Figure 6: MySQL table field description of “locations” table.

We built this table storing the different combinations of route-collectors and CCs of every year as a set of tuples in Python. Once all the combinations were collected, we created the table “locations” and we inserted the distinct route-collectors with their corresponding CC.

Let’s take a look of the content:

id	location	cc_code
1	route-collector.dac.pch.net	BD
2	optiglobe.woodynet.pch.net	US
3	route-collector.nlv.pch.net	UA
4	route-collector.nbo.pch.net	KE
5	route-collector.equinix-paris.pch.net	FR
6	200paul.woodynet.pch.net	US
7	route-collector.rno.pch.net	US
8	route-collector.cdg.pch.net	FR
9	route-collector.sin.pch.net	SG
10	nspixp2.woodynet.pch.net	JP
11	route-collector.laiix.pch.net	US
12	route-collector.bjl.pch.net	GM
13	waix.woodynet.pch.net	AU
14	equinix-ashburn.woodynet.pch.net	US
15	oix-route-views	US
16	route-collector.vie.pch.net	AT
17	amsix.woodynet.pch.net	NL
18	route-collector.sea.pch.net	US
19	route-collector.nl-ix.pch.net	NL
20	router.ams.woodynet.net	NL

Figure 7: 20 first boxes geolocated in “locations” table.

At each IXP is deployed at least one route-collector, so we had to contact PCH again to do the correct matching since the data for geolocating the route-collectors was biased. At the same time, we looked for the date of launch of each IXP and wether they are operational or not (figure 8).

IXP	IXP_name	route_collector	CC	country	city	date	operational
CINX	Capetown INternet eXchange	route-collector.cpt.pch.net	ZA	South Africa	Cape Town	20090731	Yes
CINX	Capetown INternet eXchange	router.cpt.woodynet.net	ZA	South Africa	Cape Town	20090731	Yes
JINX	Johannesburg INternet eXchange	route-collector.jnb.pch.net	ZA	South Africa	Johannesburg	19960631	Yes
JINX	Johannesburg INternet eXchange	router.jnb.woodynet.net	ZA	South Africa	Johannesburg	19960631	Yes
JINX	Johannesburg INternet eXchange	jinx.woodynet.pch.net	ZA	South Africa	Johannesburg	19960631	Yes
DINX	Durban INternet eXchange	route-collector.dur.pch.net	ZA	South Africa	Durban	20120931	Yes
DINX	Durban INternet eXchange	router.dur.woodynet.net	ZA	South Africa	Durban	20120931	Yes
KIXP	Kenya INternet eXchange	kixp.woodynet.pch.net	KE	Kenya	Nairobi	20010231	Yes
KIXP	Kenya INternet eXchange	route-collector.nbo.pch.net	KE	Kenya	Nairobi	20010231	Yes
KIXP	Kenya INternet eXchange	router.nbo.woodynet.net	KE	Kenya	Nairobi	20010231	Yes
MIX	Mozambique Internet Exchange	route-collector.mpm.pch.net	MZ	Mozambique	Maputo	20020731	Yes
MIX	Mozambique Internet Exchange	router.mpm.woodynet.net	MZ	Mozambique	Maputo	20020731	Yes
CAIX	CAiro Internet eXchange	route-collector.cai.pch.net	EG	Egypt	Cairo	20020531	Yes
CAIX	CAiro Internet eXchange	router.cai.woodynet.net	EG	Egypt	Cairo	20020531	Yes
MIXP	Malawi IXP	route-collector.blz.pch.net	MW	Malawi	Lilongue	20081231	Yes
MIXP	Malawi IXP	router.blz.woodynet.net	MW	Malawi	Lilongue	20081231	Yes
SIxP	Sudan Internet exchange Point	route-collector.krt.pch.net	SD	Sudan	Khartoum	20081231	Yes
SIxP	Sudan Internet exchange Point	router.krt.woodynet.net	SD	Sudan	Khartoum	20081231	Yes
TunIXP	Tunisia Internet eXchange Point	route-collector.tun.pch.net	TN	Tunisia	Tunis	20111231	Yes
TunIXP	Tunisia Internet eXchange Point	router.tun.woodynet.net	TN	Tunisia	Tunis	20111231	Yes
IBIXP	Ibadan Internet eXchange Point	route-collector.ibn.pch.net	NG	Nigeria	Ibadan	20020331	No
IBIXP	Ibadan Internet eXchange Point	router.ibn.woodynet.net	NG	Nigeria	Ibadan	20020331	No
NIXP	Internet Exchange of Nigeria	route-collector.los.pch.net	NG	Nigeria	Lagos	20070531	Yes
NIXP	Internet Exchange of Nigeria	router.los.woodynet.net	NG	Nigeria	Lagos	20070531	Yes

Figure 8: MySQL table with African IXPs involved in PCH dataset.

At this point, we are able to start parsing the PCH dataset. The parsing process will be deeply detailed in the next chapter.

Chapter 4: Data parsing and storage

This chapter details the parsing process of the PCH dataset and the database provisioning. Additionally, we describe the RIRs database schema required for this project.

4.1 Parsing PCH dataset

We first take a look at the different formats in the documents of our dataset:

Content of PCH files

```
BGP table version is 51066, local router ID is 206.220.228.145
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  58.28.0.0/16     192.203.154.108                        0 9560 17435 i
*>                  192.203.154.108                        0 9560 17435 i
*  60.234.0.0/16    192.203.154.67                         0 9560 17746 i
*>                  192.203.154.67                         0 9560 17746 i
*  60.234.68.0/24   192.203.154.67                         0 9560 17746 24466 i
```

Figure 9: First example of a PCH file content.

```
BGP table version is 1322, local router ID is 206.220.228.5
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.36.148.0     204.61.210.5                           0 3856 8674 29216 i
*                   204.61.210.5                           0 3856 42 8674 29216 i
*> 199.7.77.0       204.61.210.11                          0 3856 32978 i
*                   204.61.210.1                           0 3856 42 32978 i
*> 204.61.216.0/23  204.61.210.3             0             0 42 i
*>i206.220.228.0/22 206.220.228.6           0    100       0 i
```

Figure 10: Second example of a PCH file content.

```
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  0.0.0.0/0        96.4.0.55                0      0      0 11686 2914 i
*  2.0.0.0/16       157.130.10.233           0      0      0 701 6453 12654 i
*  2.0.0.0/16       167.142.3.6              0      0      0 5056 3356 6453 12654 i
*  2.0.0.0/16       129.250.0.11           342      0      0 2914 12654 i
```

Figure 11: Third example of a PCH file content.

```
##############################################################################

route-views.oregon-ix.net>term len 0
route-views.oregon-ix.net>sh ip bgp sum
BGP router identifier 198.32.162.100, local AS number 6447
BGP table version is 603151, main routing table version 603151
126616 network entries and 6124107 paths using 257246032 bytes of memory
1063261 BGP path attribute entries using 63802740 bytes of memory
821667 BGP AS-PATH entries using 22320456 bytes of memory
5549 BGP community entries using 226278 bytes of memory
3 BGP extended community entries using 72 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
Dampening enabled. 4686 history paths, 4717 dampened paths
BGP activity 155228/130058234 prefixes, 7164649/999169 paths, scan interval 15 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
4.0.4.90        4     1  124201    3803   603141    0    0 2d15h      118535
62.164.11.10    4  8782    5904    3802   603141    0    0 2d15h        2810
BGP table version is 603152, local router ID is 198.32.162.100
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          From             Reuse    Path
*d 205.210.30.0     193.0.0.56       00:00:10 3333 3356 3561 852 i
*d 80.253.192.0/20  141.142.12.1     00:00:10 1224 38 2914 3549 1759 21482 21482 21482 i
*d 209.150.90.0     216.140.14.186   00:00:10 6395 4323 11636 i
*d 206.169.96.0     216.140.14.186   00:00:10 6395 4323 11636 i
```

Figure 12: Fourth example of a PCH file content.

Table 11: Examples of PCH files.

As we can see, figures 9, 10 and 11 have pretty much the same content format. However, not every parameter has a valid value (e.g. parameters with a blank). Moreover, in these examples we can see that the beginning of the valid data is different in each case, so specific parsing will be needed for each data source.

We started removing all the escape sequences (e.g. '\n') and the information before the 'Network' word. We also suppressed non-valid characters before splitting the content of each line (e.g. 'd' and '>'). Besides, some lines in the files started without a network (see figure 9 and 10), so we had to keep the previous network value for those ones. However, when parsing the files with the format shown in figure 12, many issues appeared since the format changed completely. This problem was solved with a script developed by the research team. The result of all the development dealing with the formatting issues was a method called *parse*. It will be represented with a black box in the next section.

Once we got rid of the file content issues, we defined the table structure that stores this data (see figure 13).

Field	Type	Null	Key	Default	Extra
url_line	varchar(300)	YES		NULL	
id	int(11)	NO		NULL	
date	datetime	NO	PRI	NULL	
time	float	NO		NULL	
year	varchar(4)	NO		NULL	
location	varchar(255)	NO	PRI	NULL	
network	varchar(50)	NO	PRI	NULL	
nh	varchar(30)	YES		NULL	
nextas	varchar(30)	YES		NULL	
metric	int(11)	YES		NULL	
locprf	varchar(10)	YES		NULL	
weight	int(11)	NO		NULL	
as_path	varchar(5000)	YES		NULL	
origin	varchar(12)	YES		NULL	
as_path_length	int(11)	NO		NULL	
cc_code	varchar(2)	YES		NULL	

Figure 13: MySQL table field description of <Continent>_BGPdata_<year> .

Most of these parameters of the <Continent[2]>_BGPdata_<year> table are extracted directly form the file name and its content (further details are explained in section 4.2). For example, the *date, time* and *location* fields are extracted from the file name. However, the *url_line* field stores the path in the server towards the document and the parsed line number.

Note that *nh* field refers to the Next-Hop attribute and *origin* to the AS that sends the data ('i' or 'IGP' if it is an internal AS, 'e' or 'EGP' for an external AS and '?' or 'INCOMPLETE' for not known ASes). Finally, *id* will match with the *id* value in "locations" table.

4.2 Storage algorithm

In this section we will describe the algorithm that allows us to store the data in the database and the difficulties found in the process.

First, we build a dictionary called *downloaded_data* that contains a summary of the downloaded data in a year. The keys are the route-collector names and the values are the list of file names of a route-collector. Those names in general are under the format "<route-collector name>.<date>.gz".

Nevertheless, before starting to parse the full dataset, we need to keep track of the files parsed. Therefore, we created a document called *files_inserted_<region>_<year>.txt* where all the file names that were completely inserted will be registered. Then, we check whether that file exists before start parsing and in that case, we create a list called *files_parsed*. This file is also useful in cases where the connection to the database is lost.

[2] In the whole thesis, we follow a five region/continent division (see figure 1).

As in previous cases, we need to connect Python with the database and create a cursor object in order to store the results. Since we already geolocated all the route-collectors in the "locations" table, we build the dictionary *location_id_cc* that allows us to fill the parameters *id* and *cc_code* at each route-collector when extracting the route-colllector name of each file (see figure 14). In order to do so, we build the dictionary *location_id_cc*, fetching those parameters from the database. The key of the dictionary is the route-collector name and the value is a list of data entries. The first data in that list is the id number and the last one is the CC.

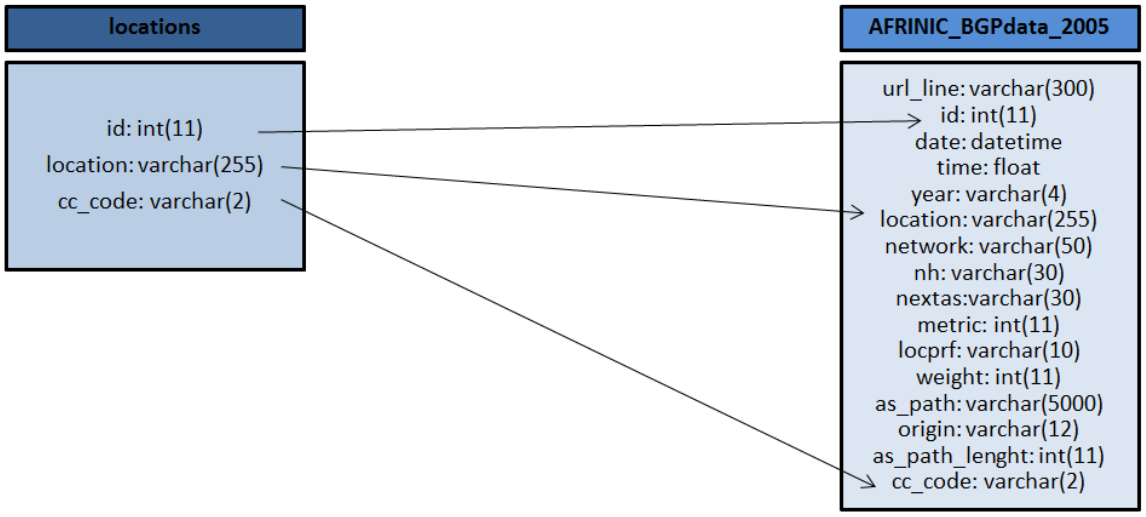


Figure 14: MySQL <Continent>_BGPdata_<year> table relation with "locations" table.

We further describe the algorithm with a flowchart (see figure 15). Note that we need to also retrieve as input data the continent, the year under study, and a dictionary called *continent_cc* with the regions as keys and the list of 2-letter CCs as values. The dictionary *continent_cc* is built with the information downloaded from [62].

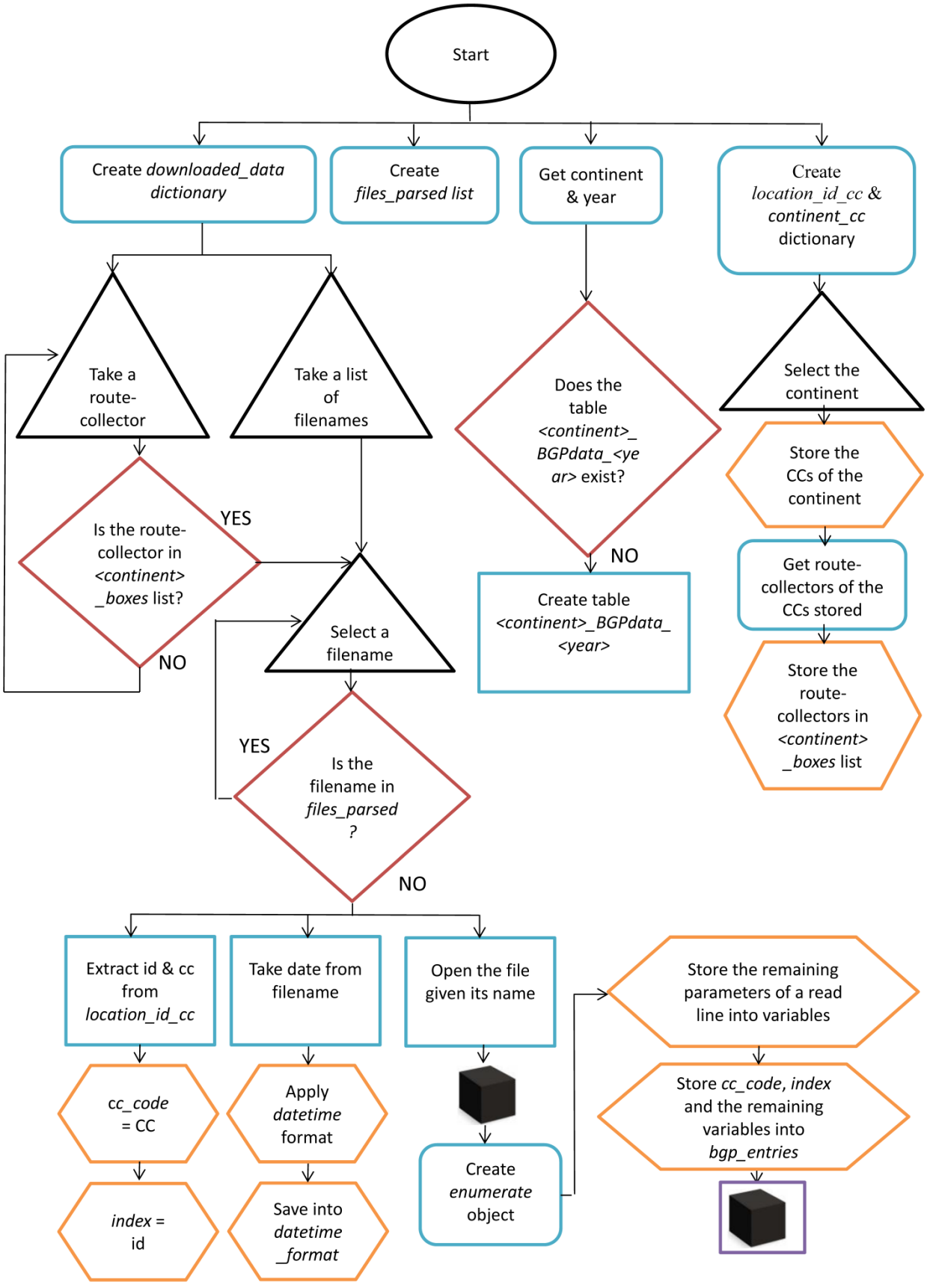


Figure 15: Storage date algorithm in *<Continent>_BGPdata_<year>* table.

We first create the *<Continent>_BGPdata_<year>* table given the continent and the year under study. Secondly, we extract in a list called *<continent>_boxes* the names of the route-collectors in the continent under study. Subsequently, we iterate over the dictionary *downloaded_data*, with the Python method *iteritems()* [63]. This method returns the key (which is the location name) and the list of files together as parameters. Next, we check whether the route-collector selected is in the continent under study. In that case, we take a filename and we check whether it is already parsed, in such case we continue looking for a non-parsed filename.

When a non-parsed file is found, we extract the *cc_code* and the *id* parameters from the *location_id_cc* dictionary given the route-collector name. We extract the date from the filename and store it as a *datetime* object into the variable *datetime_format*. We also stored a *timestamp* parameter (in the database as a *float*) whose name is *time* in the database. This step is not included in the flowchart due to space issues so the figure clarity is not compromised.

Then, we open the file, apply the parsing script developed by the research team, and with the output we create an *enumerate* object [63]. This object returns two variables; the first one is the entry number (*entry_n*) and the read line (*bgp_entry*). The line is returned as a list of the different elements on it. The path is also considered as an element in the line, which avoids collecting all the ASes before the next BGP parameter.

Next, the remaining BGP parameters are stored in variables. Note that *url_line* field stores the path in the server towards the document and the parsed line number (*entry_n)*. Since our method is recursive, in order to open each file we need to be located at the immediately superior level on the server path towards it. Consequently, the path is stored in variable where the filename is added each time a file is opened, and then we complete the url_line data adding to that path the parsed line number as "_<*bgp_entry*>".

However, the script given by the research team sometimes returned a blank or a null value when a parameter was not valid. Hence, when any of those cases was found we had to apply the formatting required for the parameter in the database. For instance, if an empty path was found, we needed to assign a 0 value to the variable *as_path_length* and a 'NULL' value to the *nextas* variable. Those special cases also apply to the *locprf*, *next_hop*, and *metric* BGP parameters. Besides, in order to match the *origin* parameter with the RIR database data, we needed to change the values 'i', 'e', and '?' to 'IGP', 'EGP', and 'INCOMPLETE' respectively.

Once we have all the parameters of a line correctly parsed, we can store them in *bgp_entries.* Initially, *bgp_entries* was a list containing the parsed data in a defined order of insertion. But we improved the performance of this script by executing batch "INSERTs" in the database. In order to do so, the new *bgp_entries* is a list of lists. Of course, we ignore the parsed lines that are exactly the same.

Then, we compute the length of the new *bgp_entries* list. We insert a hundred parsed lines together. But, if for some reason there are less parsed lines in the list (e.g. the file has ended before

reaching a length of 100), we insert each list in *bgp_entries* one by one. Finally, when we have inserted all the data of a file, we append the filename to *files_inserted_<region>_<year>.txt* and to the *files_parsed* list (this whole process is represented as a black box [64] inside a purple rectangle in figure 15).

For the purpose of this algorithm, all the lists used could be sets instead, except the *bgp_entries* list due to the fact that the parameters should be in a defined order so we can insert them in the database together.

Regarding the difficulties found, it took too much effort to configure the insertion of multiple queries in MySQL, since its version on the server was really restrictive. On the other hand, as mentioned in section 2.2.3.1, we needed an expansion to 4TB on the server. All the data regarding the AFRINIC region, takes around 2 GB. However, since we parsed all the dataset for future work purposes, the complete dataset occupies as much as 600 GB. For instance, some files from RIPE NCC weigh more than 1 GB on a given date.

4.3 RIRs database

PCH has an open peering policy. It does not cover all the information of the peers of all the IXPs. So there it lacks some data, despite we have a good approach in comparison with RouteViews [5]. Since we want to analyze how African prefixes and Autonomous Systems (ASes) are seen from other countries, we will also need the information contained in the RIRs (Regional Internet Registry) database.

Note that this database was provided by the research team. Since we had to update it, we can talk about its structure and content. The data of these RIRs is stored at a database with the following structure per region (see figure 16).

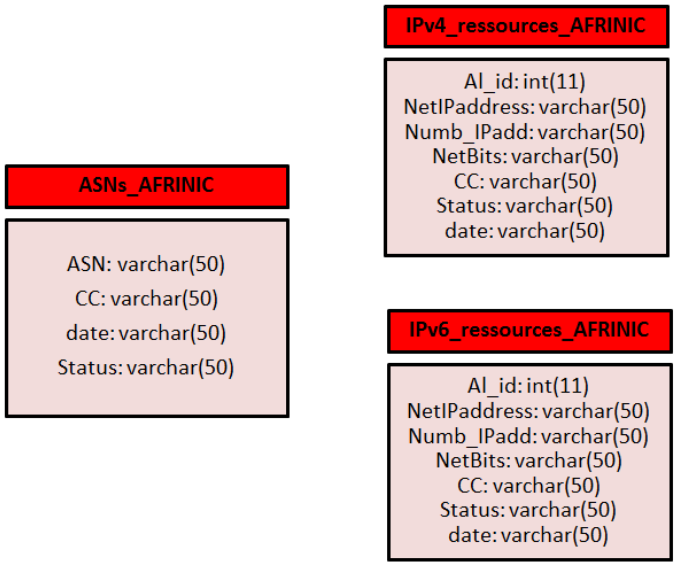


Figure 16: Relational RIRs database structure for African information.

Initially, this database had 15 tables. There are three different tables per region, one for the Autonomous System information and two more for the prefixes, depending whether they are in IPv4 or in IPv6 format. All the tables have in common that for every element there are at least three other parameters to characterize them: date, status and CC.

It is appreciated that the status defines if a prefix or an AS is assigned, allocated, reserved, returned or available at the region. Therefore, we will study the elements that are allocated or assigned at each region, as they allow us to distinguish the relevant data. Moreover, at the tables the prefixes are separated into their network address number and their mask into two parameters: *NetIPaddress* and *Netbits* respectively. The reason why they are separated is that the mask was not directly extracted from the files downloaded [65] when building this database, and they had to be computed with the number of IP addresses stored at *Numb_IPadd* in the database.

Lastly, we included another table with the IXPs information afterwards. This table was built with the information at table 1, the route-collectors at PCH and the previous tables at AFRINIC region in RIR database.

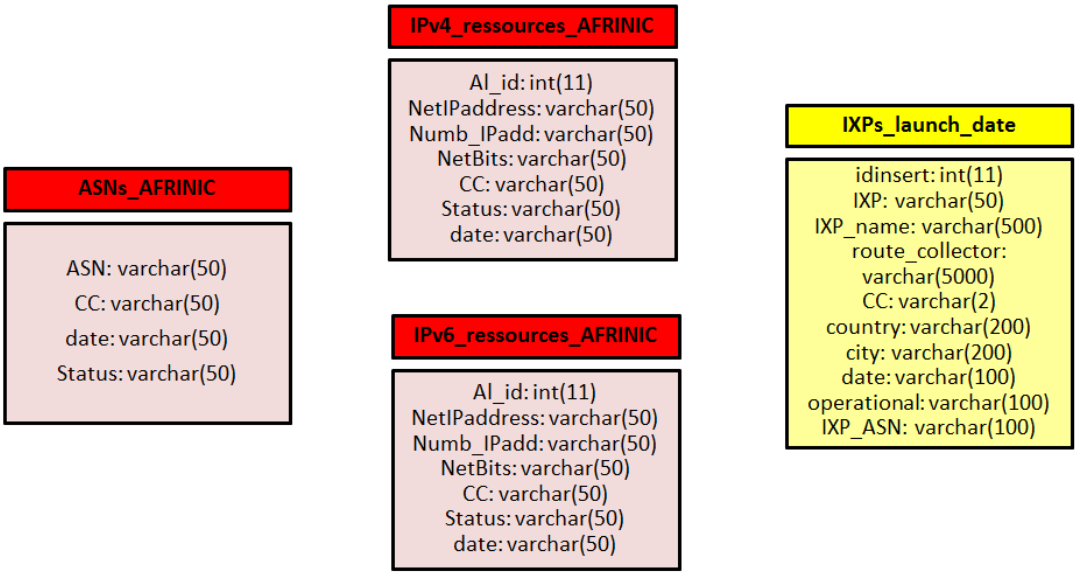


Figure 17: Complete relational RIRs database structure for African information.

Note that in this new table, the id per entry is not related with the other table ids as well as the date parameter only stores the date when the IXP was set as operational in the format:

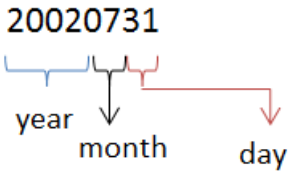


Figure 18: Date format in RIRs database.

AFRINIC was launched in 2004 [66]. Consequently, we noticed that in the delegated PCH file, prefixes and ASNs attributed before this year sometimes correspond to biased allocation dates. This fact also influenced our choice perform this work from 2005 on. Finally, this database occupies 89 MB on the server.

Chapter 5: Statistics (part I)

This chapter details the first studies developed for achieving the objectives of this thesis. The remaining statistics are dependent of these studies (since we reuse some results) and therefore they are detailed in the next chapter. For each one of them, there is an introduction, a detailed explanation of the algorithms required, a description of extra resources if needed, a graph or table with the results, and a brief discussion about them.

It is important to keep in mind table 12 for keeping track of the explanations given along chapters 5 and 6. It is also important to consider that all the IXPs in Africa are not covered by the dataset. Besides, PCH route-collectors are not deployed as soon as IXPs are launched. Although PCH has an open peering policy, not all ISPs at an IXP peer with PCH boxes. Finally, it is important to take into account that for some prefixes allocated by other RIRs, the allocation dates in the AFRINIC delegated files are biased (e.g. 00000000, years 1984, 1989, 1990, etc.).

CC	Country	Cities	Route-collectors	IXP	Date of launch IXP
ZA	South Africa	Cape Town	route-collector.cpt.pch.net router.cpt.woodynet.net	Capetown Internet Exchange (CINX)	July 2009
		Johannesburg	route-collector.jnb.pch.net router.jnb.woodynet.net jinx.woodynet.pch.net	Johannesburg Internet Exchange (JINX)	June 1996
		Durban	route-collector.dur.pch.net router.dur.woodynet.net	Durban Internet Exchange (DINX)	September 2012
KE	Kenya	Nairobi	kixp.woodynet.pch.net route-collector.nbo.pch.net router.nbo.woodynet.net	Kenya Internet Exchange Point (KIXP)	February 2001
MZ	Mozambique	Maputo	route-collector.mpm.pch.net router.mpm.woodynet.net	Mozambique Internet Exchange (MIX)	July 2002
EG	Egypt	Cairo	route-collector.cai.pch.net router.cai.woodynet.net	Cairo Internet Exchange (CAIX)	May 2002
MW	Malawi	Lilongüe	route-collector.blz.pch.net router.blz.woodynet.net	Malawi IXP (MIXP)	December 2008
SD	Sudan	Khartoum	route-collector.krt.pch.net router.krt.woodynet.net	Sudan Internet Exchange (SIxP)	October 2011
TN	Tunisia	Tunis	route-collector.tun.pch.net router.tun.woodynet.net	Tunisian Internet Exchange (TunIXP)	2011
NG	Nigeria	Ibadan	route-collector.ibn.pch.net router.ibn.woodynet.net	Ibadan Internet Exchange (IBIXP)	March 2002 (no longer operational)
		Lagos	route-collector.los.pch.net router.los.woodynet.net	Internet Exchange of Nigeria (NIXP)	May 2007

Table 12: African IXPs involved in PCH dataset.

Note that since IBIXP is no longer operational, we will not give results of it along 5th and 6th chapter.

5.1 Time difference between prefix Allocation date and Appearance on the Internet

The objective of this experiment is to know whether organizations receiving prefixes from AFRINIC actually make use of them. IPv4 address space is depleted, so if they see prefixes are not used for internet routing, they could consider reclaiming them.

This analysis is going to be divided into two different parts over the three subsections: algorithm explanation, extra resources needed and results. The first part will consider the time gap between the first allocation date, which is the date a network is attributed to an operator, and the first appearance on Internet, which is the date the prefix was seen on Internet. On the same graph, we will also plot the elapsed time between the last appearance date and the consultation. The second part will illustrate the difference between the last allocation date and the first appearance on Internet, as well as the time gap between the last appearance date and the consultation one. All these differences will be in months, unless stated otherwise.

5.1.1 Algorithm

In order to obtain the data for the first graph in this item, we will make use of a flowchart that is going to be broken in subparts for the explanation.

Firstly, we need three different inputs: the data stored in the database RIRs, a file with RIPE stats [65], and the consultation date (see figure 19). RIPE stats is based on Routing Information Service (RIS) collectors and it is from where we found out the first appearance date of each prefix and the last appearance date on the Internet. Note also that the last appearance date might not be the same as the consultation date.

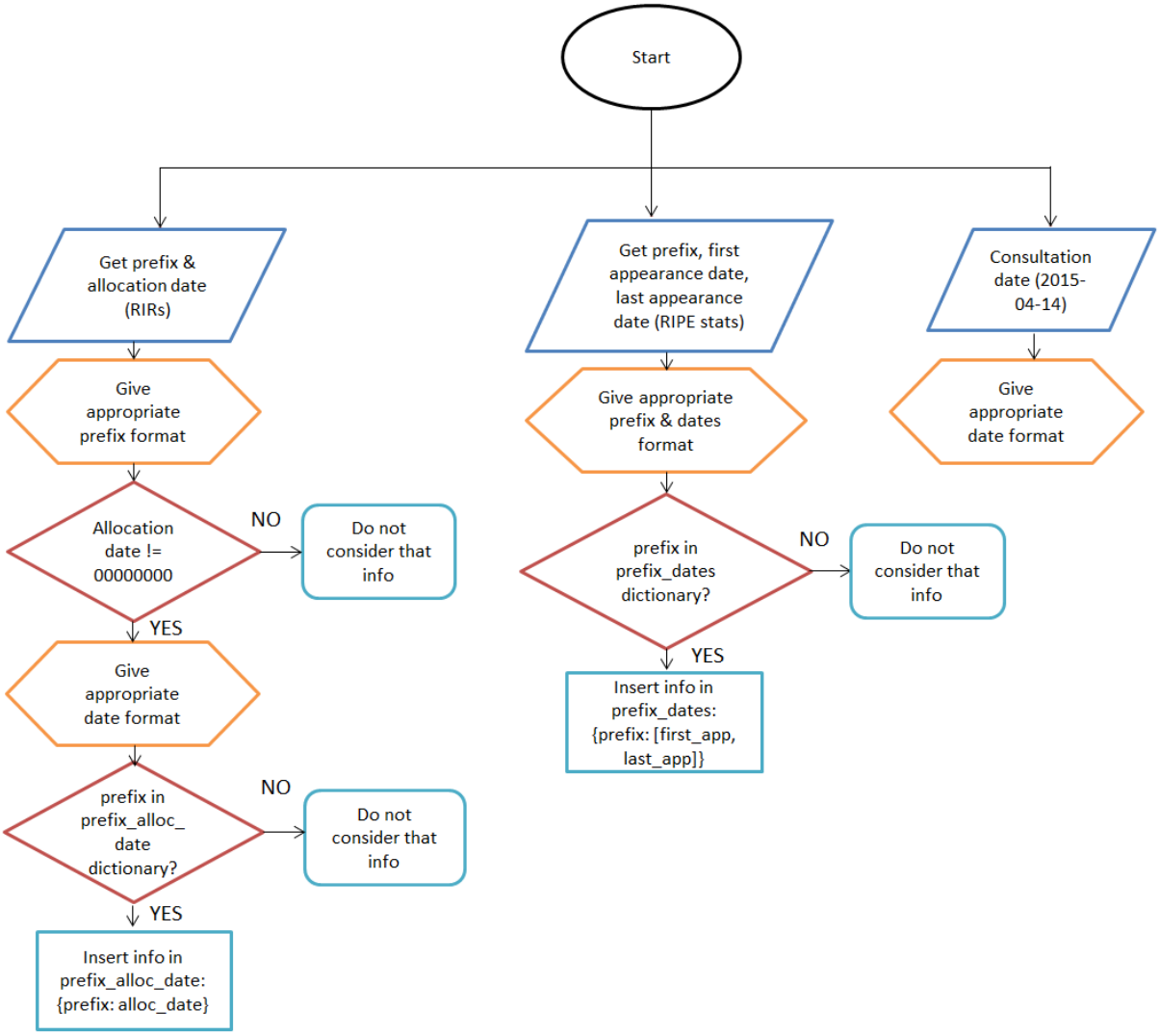


Figure 19: Descriptive flowchart with the input data formats and storage in memory.

Once the needed input data has been defined, we will proceed to describe how the time between allocation and appearance on Internet is computed, and the months difference between the last appearance date and the consultation one (shown in figure 19).

We extract from the *RIRs* database the allocated or assigned prefixes. The query will return the distinct parameters which contain the prefix and allocation date information ordered by ascending date. The prefix was stored into two parameters at the database, one with the prefix number and the other one with the subnet mask. Consequently, we had to join that information with a slash ('/') in between to match with the format given in RIPE stats. Similarly, we had to give the appropriate format for the dates, so that we could perform a *datetime* subtraction later. We defined it as "day-month-year hour:minutes:seconds". But before doing so, we must check whether the default date is given (*00000000*), since for some prefixes we did not get the allocation date so we gave them a default date instead. In this case we cannot perform a valid subtraction and the prefix under that date will not be considered. After formatting correctly the data we store that

information in a dictionary structure, checking before inserting whether the prefix was already stored, since the first time we return a prefix we will obtain the first time appearance. Therefore, the key of that dictionary will be the prefix, and the value will be the allocation date associated to that network.

Additionally, we will need another dictionary that will store the RIPE stats data. As we did before, we firstly apply the correct format to the data (*string* and *datetime* types), and then store it in a dictionary where the keys will be the distinct prefixes found previously, and the value for each one of them will be a list where the first element will be the first appearance date, and the second element will be the last appearance date.

The last input data we need is the consultation date, which was April 14th 2015 stored as a *datetime* variable, so that we could reuse the same code.

We will continue the description of the algorithm with another flowchart, reduced to our purposes (see figure 20). Note that the black boxes represent the previous flowchart processes for building the *prefix_alloc_date* and *prefix_date* dictionaries, and applying the correct date to the consultation date for accomplishing our purposes.

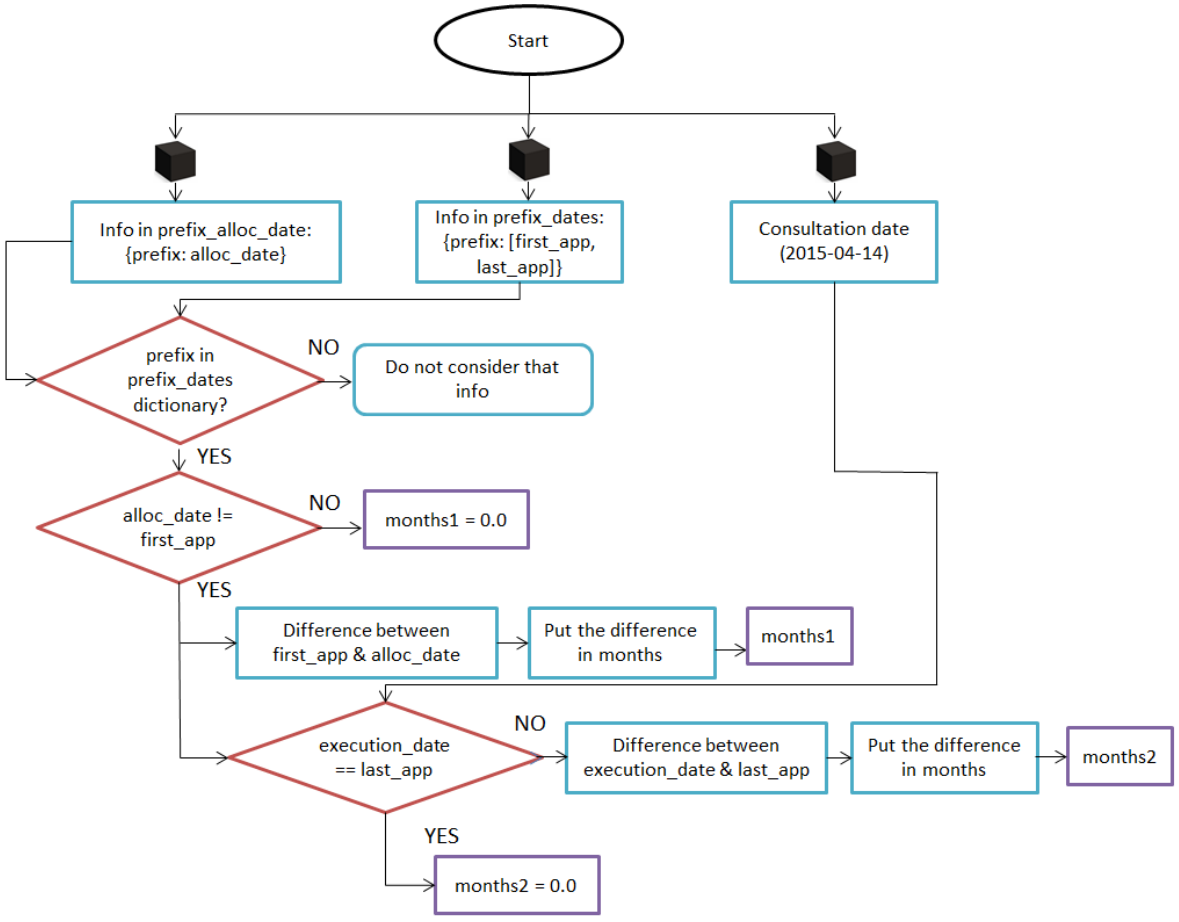


Figure 20: Descriptive flowchart with the operations over the input data.

After storing in memory the information with the desired format previously mentioned, we take one by one the prefixes in the dictionary *prefix_alloc_date* and we check if it is in the dictionary *prefix_dates*. If this is not the case, we will not consider this information.

However, if a prefix is present in both dictionaries, we can compute the differences between the first appearance date and the allocation date (whose result will be stored in a variable called *months1*) and the difference between the consultation date and the last appearance date (stored in a variable called *months2*).

It might happen that the first appearance date and the allocation date are the same, in such a case we directly assign 0.0 as the result of the difference. Similarly, it might occur that the consultation date and the last appearance date coincides, so the result of the difference that we assign is also 0.0.

The final step is to keep both time deltas, such that we could represent them graphically. To visualize the results, we build a dictionary with the prefixes as keys and the months differences in a list as value. We are interested in saving the information in order, so that we will order the dictionary for our purposes by value. As a result, we got two different files with the following format:

- File 1: Prefix, first appearance date and the allocation date (*months1*).
- File 2: Prefix, consultation date and the last appearance date (*months2*).

We have also computed another graph with the months difference between last allocation date and the first appearance, and also the months difference between the same consultation date and the last appearance date. The script is pretty similar to the one written for the other graph, changing just the order of the parameters returned by the first query.

Regarding the implementation of the algorithm, it is possible to use another datatypes, or store the information not in dictionaries but in hash tables. We decided to use *datetime* instead of *date* format, since it could happen that the appearance date and allocation date was the same and then, the difference in days will be 0 without taking into account the time. We also decided to work with dictionaries in order to reduce the code for storage as well as for accessing the data in comparison to hash tables, since hash tables need additional checks for inserting data and additional loops for extracting data from them. Moreover, we could easily order the dictionaries as we want (by keys or by values) just with a coding line.

5.1.2 Extra resources needed

MATLAB need special formats for plotting the results, especially when characters like a slash (‘/’) are involved. Furthermore, if files contain huge amounts of data, is better to give several files instead of just one.

Therefore, we had to assign a reference number to each prefix in order to have a clear graph. Plus, we had to separate the information under study in two different sources so MATLAB could match the information correctly and we did not have false or null matches.

Another consideration we had to be careful about, was the way the information is structured in the file, as the reading functions expects the data in a specific format for which special characters could be mistaken. For example, characters like ';' would stop the reading of a file before reaching the end of it.

We also programmed a script which returns as output the number of prefixes per CC that have appeared at the same date than the allocation date and the number of prefixes that appear after a year of its allocation date. In order to do so, we reused the script and we established as a value to the dictionary *prefix_alloc_date* a list with two elements ([*allocation date, country code*]), instead of just the allocation date. The CC was taken from the database when the prefix is also extracted. Then, the dictionary *prefix_dates* will have an additional parameter as a value which is the CC assigned to the prefix.

Next, we stored into two different dictionaries the prefixes seen at a country depending if the months differences calculated is equal to 0.0 or greater than 12.0. Therefore, the keys of the dictionaries will be the CCs and the values a unique list of prefixes given a specific country. With this information we will provide the top four countries whose prefixes appear at the allocation date and the number of prefixes in the year after the allocation date in that country.

It is remarkable that in general Internet Assigned Numbers Authority (IANA) has granted 3,227 v4 prefixes to AFRINIC. Nevertheless, 3,067 are allocated or assigned. Some prefixes are assigned many times, so we take the first country where they have been appearing in our studies when considering the time gap between the first allocation date and the first appearance on Internet.

Finally, we also created three files for storing the prefixes and the months difference of the networks that appear in the same day of the allocation date, the ones that appear in the same year of the allocation date and the ones that do not appear in the same day but have a positive months difference. Thus, in the next section we will provide a deep analysis of the graph given the algorithm shown in the previous section.

5.1.3 Results and graphs

While treating the data, we found some prefixes that have been attributed twice by AFRINIC. For those we only consider the first allocation date. Taking this into consideration, we found about 268 prefixes reallocated. When they are, they are sometimes subnetted into prefixes of lower sizes (for instance 165.143.0.0/13 has been reallocated into a /16, etc.).

Although AFRINIC allocates a prefix, it keeps ownership over that prefix. For instance, when it notices that an operator is not using an assigned prefix, AFRINIC may reallocate the prefix to another operator. It is also possible that if two companies merge into a new legal company, they will have to re-contract and re-allocate. Hence, it would be a nice result to identify and quantify.

As we computed the difference between the first allocation date and the appearance date, we have large differences. We will add more details about what could happen in the next item.

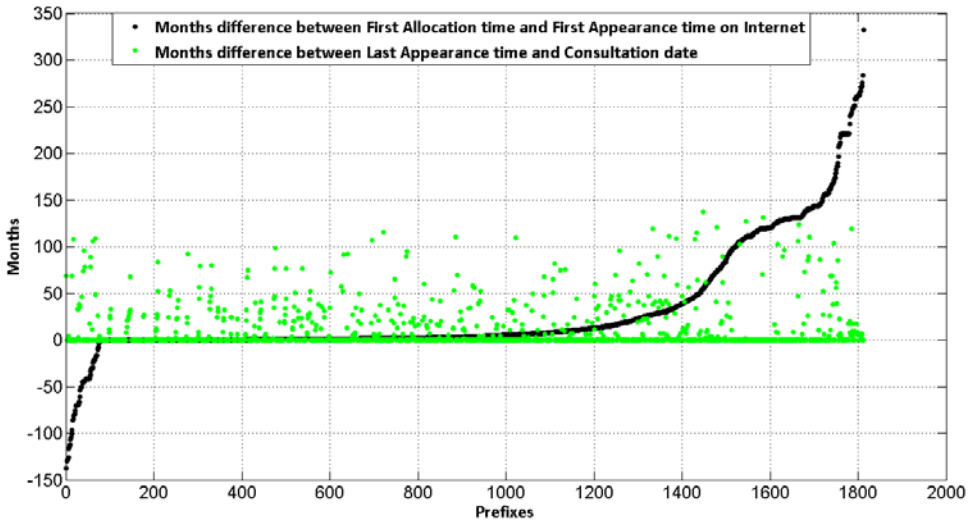


Figure 21: Months difference between the First Allocation time and the Consultation date (April 14, 2015) given RIPE stats for AFRINIC prefixes over time.

1,833 v4 prefixes out of the 3,067 have been allocated or assigned by AFRINIC to organizations in the region as of April 14th 2015. Among them, just 1,812 appear on Internet according to RIPE stats. Hence, 98.85% of allocated v4 prefixes appear on the Internet.

This graph also shows that only 25.28% of prefixes do not appear on the Internet on the consultation date. The remaining 1,354 appear at that date. Most importantly, 1,509 have appeared in 2015 as the last year of appearance, i.e. 83.28% of the prefixes under study appear in 2015.

4.42% prefixes appear before their allocation date. The minimum negative difference of these prefixes is 0.266 months (8 days) and the maximum one is 137.66 months (11.5 years). However, the two minimum positive months differences are 0.0333 (1 day) and 0.0666 (2 days), and the two maximum are 331.866 and 283.566 months (28 years and 24 years respectively). Indeed, for some prefixes allocated by RIRs, the allocation dates in the AFRINIC delegated files are biased (e.g. 00000000, 1984, 1989, 1990 years), although AFRINIC was launched in 2004 [66]. About 7.4% prefixes are in such case. We plan to search in other RIR delegated files the last allocation date in order to reduce that gap.

As we said before, we provide the top four countries to which the prefixes first appearance is the first allocation date and also the prefixes whose first appearance is after a year of the first allocation date.

CC	Number and ratio of prefixes seen on Internet at the first allocation date
ZA	228 (12.58%)
KE	101 (5.57%)
EG	86 (4.74%)
NG	67 (3.70%)

Table 13: Top four countries with highest number of prefixes appearing at the first allocation date.

CC	Number and ratio of prefixes seen on the Internet a year after the first allocation date
ZA	75 (4.14%)
NG	37 (2.04%)
EG	26 (1.43%)
GH	21 (1.16%)

Table 14: Top four countries with highest number of prefixes appearing after a year from their first allocation date.

From table left and right, we learn that prefixes allocated to Kenya organization are used either on the same day or within a year after their allocation date.

To sum up, 95.58% of v4 allocated prefixes appear on the Internet from their allocation date registered by AFRINIC. 87.12% have appeared lastly in 2015. Moreover, the prefixes that have appeared most are from South Africa, Nigeria and Egypt.

In the second part, we computed a graph showing the results for the case in which the Last Allocation date is involved as well as the Last Appearance date.

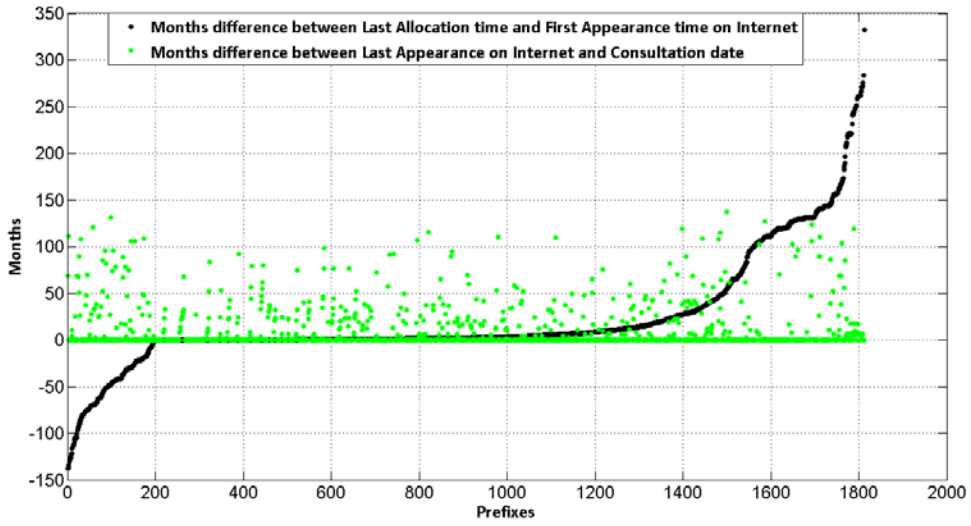


Figure 22: Months difference between the Last Allocation time and the Consultation date (April 14, 2015) given RIPE stats for AFRINIC prefixes over time.

From this graph, we observe that 25.28% prefixes whose last appearing date is the consultation one. The remaining 1354 appear at that date. Most importantly, 1,509 have appeared in 2015 as the last year of appearance, i.e. 83.28% of the prefixes under study appear in 2015. Then, 16.72% prefixes last appeared on the Internet before 2015. Although we thought this was not possible, we were able to check that the months difference changed a bit, but not significantly.

The same ratios regarding prefixes appearing before their last allocation date and appearing after a large months difference (above 40 months) were obtained in this case. Moreover, the maximum positive months difference was the same. However, the two minimum positive months differences are 0.0333 months (1 day) and 0.3666 months (11 days). The maximum negative months difference also differs, which is 138.133 months (11.5 years).

Once more, we provide the top four countries to which the prefixes first appearance is the last allocation date and the number of prefixes appearing at that country. We do the same for the prefixes whose first appearance is after a year of the last allocation date.

CC	Number and ratio of prefixes seen on Internet at the last allocation date
ZA	490 (27.04%)
KE	99 (5.46%)
EG	85 (4.75%)
NG	63 (3.48%)

Table 15: Top four countries with highest number of prefixes appearing at the last allocation date.

CC	Number and ratio of prefixes seen on the Internet a year after the last allocation date
ZA	74 (4.08%)
NG	45 (2.48%)
EG	27 (1.49%)
GH	20 (1.10%)

Table 16: Top four countries with highest number of prefixes appearing on the Internet after a year from their allocation date.

In contrast to the top CCs in table left, we notice that except for South Africa, the number of prefixes appearing at the last allocation date is slightly reduced. But, for the case of South Africa, the number of prefixes appearing at the same date than the last allocation date is more than twice the number of prefixes appearing at the first allocation date. This is an effect of a reallocation of a prefix to another ISP. Regarding the top CCs of the first allocation date, after comparing them with the ones of the last allocation date, we see that they are approximately the same number of prefixes are seen at the same countries.

In conclusion, 95.58% of the prefixes appear since their allocation date (regardless whether it was the first allocation date or the last one) and 87.12% of them have appeared on 2015 as the year last of appearance. Moreover, the most frequent prefixes come from South Africa, Nigeria and Egypt.

5.2 Time difference between prefix Allocation and Appearance in the data collected by PCH route-collectors deployed at an African IXP

This study will show two types of graphs. On the one hand, the first type will describe the months difference evolution between the first allocation date and the first appearance of AFRINIC prefixes in PCH dataset. On the other hand, the second type will illustrate the months difference evolution between the last allocation date and the first appearance date in PCH dataset. However,

we will first base our results per route-collector, then per IXP (these results need the ones obtained per route-collector), and finally we will obtain the appearance evolution at any IXP to get a general view and a deeper understanding about the Peering in AFRINIC.

5.2.1 Algorithms

5.2.1.1 Algorithm 1: Per route-collector.

At the very beginning, we had not completed the table mentioned in this chapter. Otherwise, we could directly obtain the information per IXP and at any of them. Under those circumstances, we were able just to compute the difference at each route-collector and, since this information will be useful for the IXPs, we saved it at two different tables.

The reason why we built two tables is to ensure that there will not be an empty field when returning all the information and reuse the code for both cases. The relation between the parameters from the tables in the databases and a representation of how the tables are built is shown in the next figure.

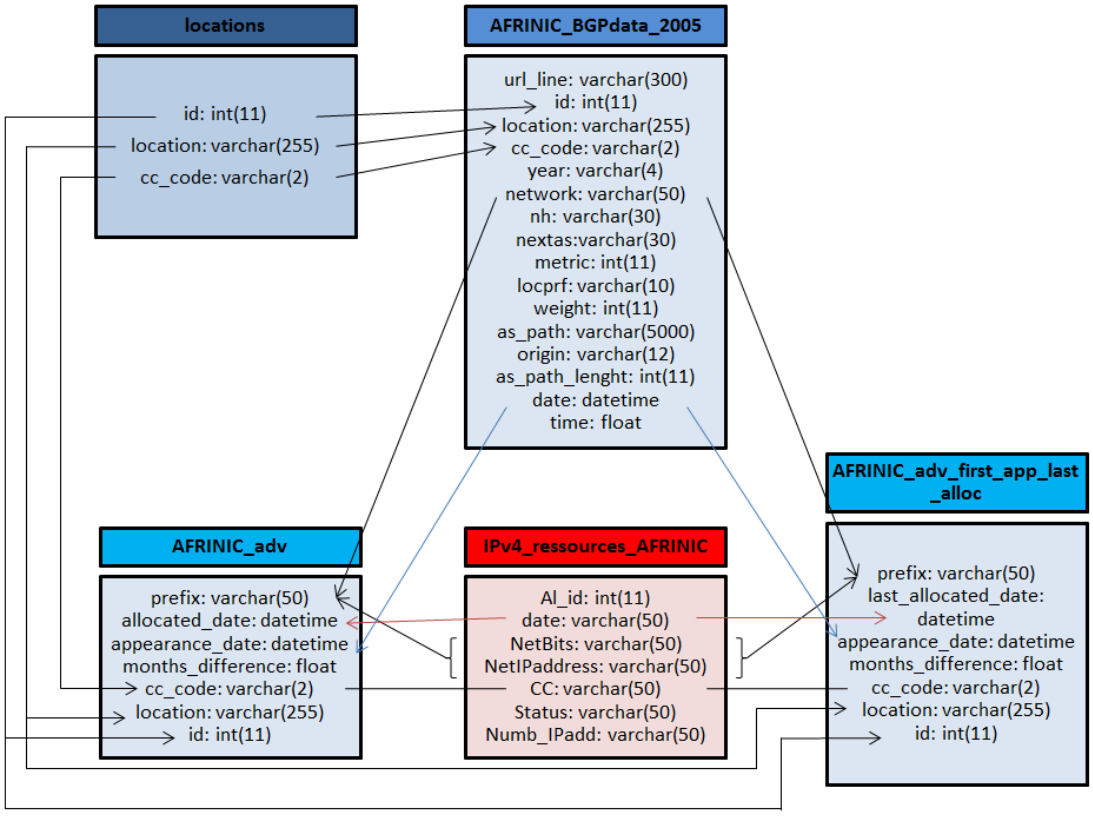


Figure 23: Implementation of tables for studying the prefix allocation and appearance.

Since the code was pretty similar for having these two tables, we just describe one of them, for instance, *AFRINIC_adv*.

Our goal is to collect the information in order to tell us whether a network connects to PCH to announce it. Therefore, the indexed fields of such table will be the network, the appearance date, the months difference between the allocated date and the appearance one, and an id referred to the route-collector and the CC assigned to it.

Once the table is created, we must create a dictionary with prefixes as a keys and the allocation date as value, as we did in the previous item. Then, we must look into the AFRINIC continent and for every year extract the remaining parameters and store them somewhere.

Again, we decided to save the information in a dictionary where the key was the network and the values were the rest of parameters joined into a structured set, in order not to have repeated those parameters mentioned, and the order will allow us to insert the parameters later easily. It was really important to have as keys the prefixes, since a prefix could be announced in to different route-collectors.

Finally, we computed the months difference for each prefix and we inserted all the parameters together.

It is true that we could have built just one table and, instead of inserting just the parameters when all of them are collected, we could have built the table first with the prefixes and the allocation date, and then add the remaining parameters found. But some prefixes may have an invalid allocation date (00000000), or even more, they are not announced by PCH, resulting in an invalid time difference that should not be taken into account in the graphs.

We avoided all those possibilities with this procedure and we reused the code with small changes for the table where the months difference is computed with the last appearance date.

An example is provided of the filled table. The negative values represent that a prefix is announced at PCH before its allocation date. We can observe, as we thought, that a prefix could be announced in to different route-collectors, as for the prefix 192.96.141.0.

prefix	allocated_date	appearance_date	months_difference	location	cc_code	id
192.96.141.0/24	2009-08-27 00:00:00	2009-01-27 00:00:00	-7.06667	jinx.woodynet.pch.net	ZA	116
192.96.141.0/24	2009-08-27 00:00:00	2009-10-01 00:00:00	1.16667	route-collector.mpm.pch.net	MZ	89
192.96.142.0/24	2001-06-05 00:00:00	2009-09-18 00:00:00	100.9	jinx.woodynet.pch.net	ZA	116
192.96.142.0/24	2001-06-05 00:00:00	2009-10-01 00:00:00	101.333	route-collector.mpm.pch.net	MZ	89
192.96.142.0/24	2001-06-05 00:00:00	2009-10-24 00:00:00	102.1	route-collector.cpt.pch.net	ZA	26
192.96.147.0/24	1993-03-30 00:00:00	2009-01-01 00:00:00	191.867	jinx.woodynet.pch.net	ZA	116
192.96.15.0/24	1992-03-11 00:00:00	2009-01-01 00:00:00	204.667	jinx.woodynet.pch.net	ZA	116
192.96.15.0/24	1992-03-11 00:00:00	2009-10-01 00:00:00	213.767	route-collector.mpm.pch.net	MZ	89
192.96.15.0/24	1992-03-11 00:00:00	2009-10-24 00:00:00	214.533	route-collector.cpt.pch.net	ZA	26
192.96.150.0/24	1993-03-30 00:00:00	2009-01-01 00:00:00	191.867	jinx.woodynet.pch.net	ZA	116
192.96.150.0/24	1993-03-30 00:00:00	2009-10-01 00:00:00	200.967	route-collector.mpm.pch.net	MZ	89
192.96.150.0/24	1993-03-30 00:00:00	2009-10-24 00:00:00	201.733	route-collector.cpt.pch.net	ZA	26
192.96.177.0/24	2009-08-27 00:00:00	2009-01-01 00:00:00	-7.93333	jinx.woodynet.pch.net	ZA	116
192.96.177.0/24	2009-08-27 00:00:00	2009-10-01 00:00:00	1.16667	route-collector.mpm.pch.net	MZ	89
192.96.177.0/24	2009-08-27 00:00:00	2009-10-24 00:00:00	1.93333	route-collector.cpt.pch.net	ZA	26
192.96.193.0/24	2002-10-25 00:00:00	2009-01-01 00:00:00	75.3333	jinx.woodynet.pch.net	ZA	116
192.96.193.0/24	2002-10-25 00:00:00	2009-10-01 00:00:00	84.4333	route-collector.mpm.pch.net	MZ	89
192.96.193.0/24	2002-10-25 00:00:00	2009-10-24 00:00:00	85.2	route-collector.cpt.pch.net	ZA	26
192.96.194.0/24	2002-10-24 00:00:00	2009-01-01 00:00:00	75.3667	jinx.woodynet.pch.net	ZA	116
192.96.194.0/24	2002-10-24 00:00:00	2009-10-01 00:00:00	84.4667	route-collector.mpm.pch.net	MZ	89

Figure 24: Example of the contents at *AFRINIC_adv* table.

5.2.1.2 Algorithm 2: Per IXP.

Now all the information per route-collector has been computed. So what we need afterwards is to save, per IXP, the distinct prefix number and the months difference of each route-collector in a document valid for plotting tools. The process followed is described in the next flowchart.

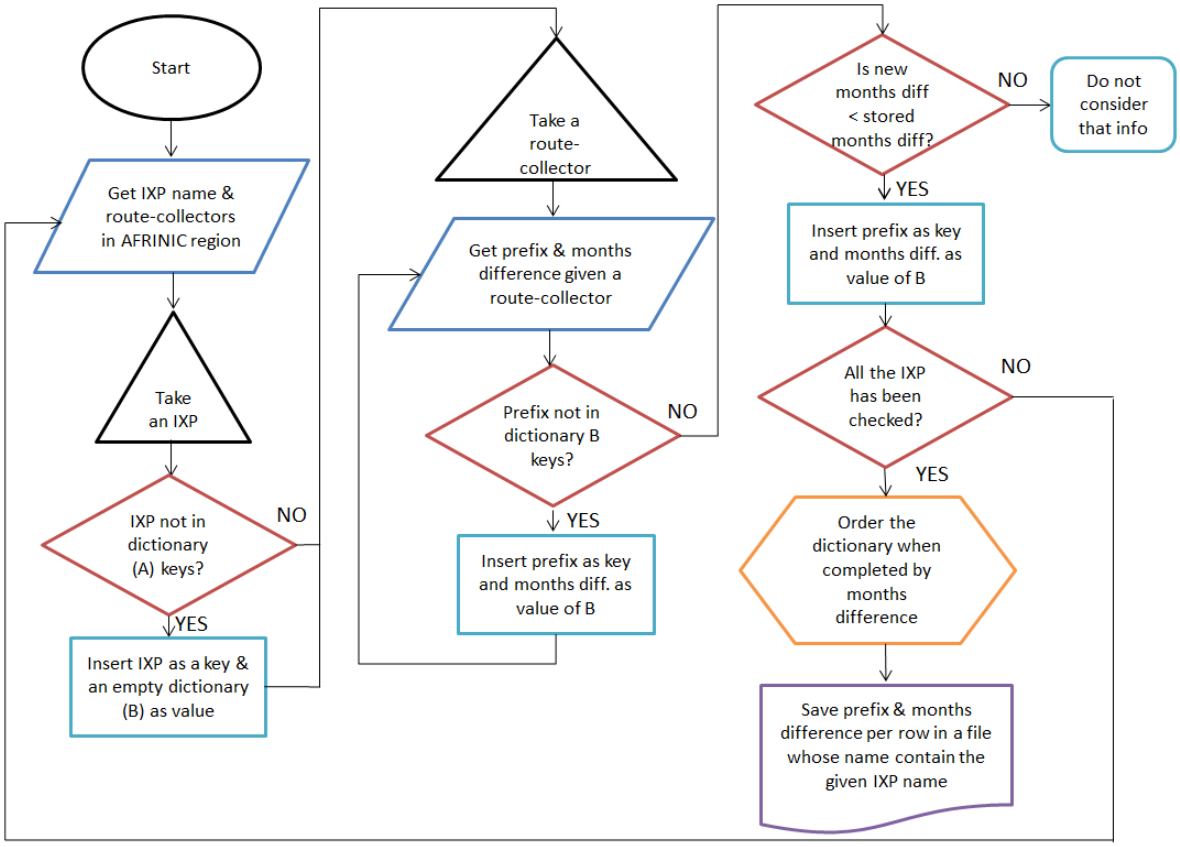


Figure 25: Flowchart for developing the prefix allocation and appearance per IXP.

The resultant documents of the prefixes announced at PCH data with respect to those from RIR were really useful for getting the general announcement of prefixes in AFRINIC region.

5.2.1.3 Algorithm 3: At any IXP.

Similar to the first experiment, we traverse a loop where all the IXPs will be considered and we extract the information of the document created with the previous algorithm. We must be careful with the escape sequences (e.g. '\r\n') when reading the data and we must also check if a prefix has two different months difference (as we said, it could be announced in two different route-collectors and hence, in two distinct IXPs) and keep the lowest months difference for our purposes.

Finally we sort the dictionary by months difference, and we store the prefixes and the differences in a file for being able to see the evolution of Peering in PCH data at any IXPs in the AFRINIC region.

5.2.2 Extra resources and comments

When developing the last algorithm in this item, there was not a file per IXP since some of the prefixes announced at PCH did not correspond to any stored, such as, in IBIXP or TunIXP. So that, we had to avoid those IXPs when searching the files to be read. We had also to remove the escape sequences for the script that plots in MATLAB the results.

As before, we assigned a reference number to each prefix for not having a messy graph. We will also provide some graphs per route-collector, as they are quite interesting and provide a deepest understanding of the results.

One of the main difficulties of these plots, besides reading the data from a file format, was to create an appropriate visualization for showing them in this document. In the same way, we had to resize the flowchart and the explanation in order to fit an approximately equal distribution of pages for each experiment.

5.2.3 Results and graphs (part I)

First, we take a look at the announcement information at any IXP in PCH dataset.

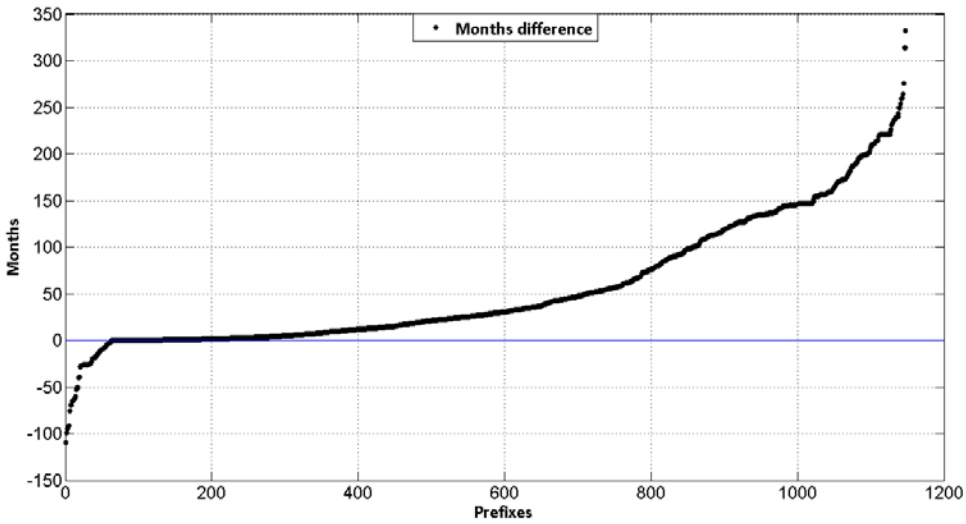


Figure 26: Months difference between the first allocation time and the first time appearance of prefixes at any IXP in the AFRINIC region.

At PCH boxes, we observe that 1,148 is the amount of IPv4 prefixes announced at any IXP in the AFRINIC region (see figure 26). Thus, the percentage of prefixes announced by ISPs that are peering with PCH boxes at African IXPs is 63.36%. In contrast with the information in RIPE stats, we

have just a prefix whose months difference between the first allocation time and the first time appearance is 0.0.

Even though the minimum and maximum positive months differences match when talking about the negative ones we have found that the maximum months difference is 109.133 months (10 years) and the minimum is 0.066 months (2 days).

From the graph, it is easy to detect that more than a half of the prefixes are announced for the first time after a year of their allocation date. Just 28.92% prefixes shown are announced in the same year of the allocation date and 5.75% of the prefixes in the graph are announced before their allocation date.

Let's take a look now at the graph which represents the evolution of months differences between the first allocation date and the last appearance date of prefixes announced at PCH route-collectors.

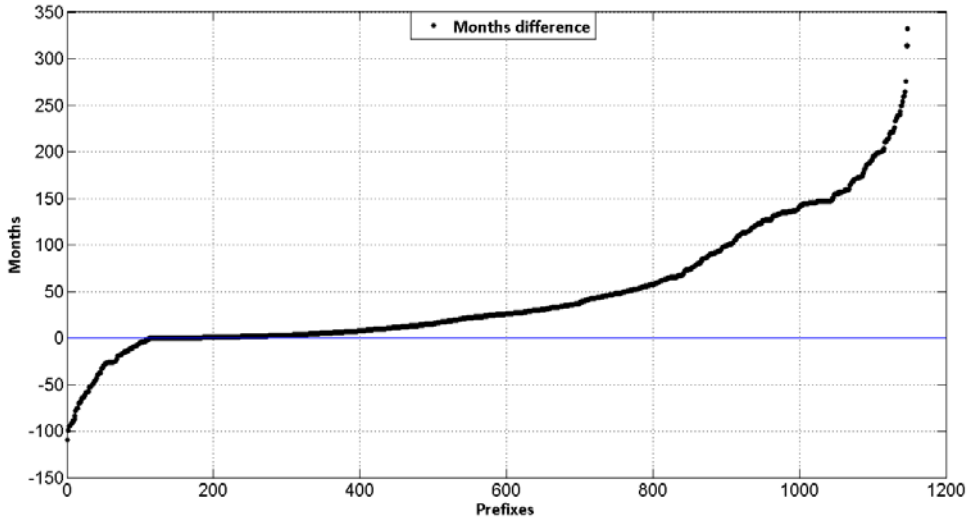


Figure 27: Months difference between the first allocation time and the last time appearance of prefixes at any IXP in the AFRINIC region.

As in the previous case, we found just 63.36% prefixes that are announced at PCH boxes. We also have the same maximum and minimum months differences for both, the negatives and the positive ones. Nevertheless, no prefix showed a 0.0 months difference.

It is clear that almost no prefixes have been appearing in an IXP before its allocation by AFRINIC. However, some are allocated by another RIR and later AFRINIC has taken over those prefixes. We know that at dates like 1981, PCH was not available and then, differences like -100 months (-9 years) or over 150 months (12.5 years) are given due to this fact. We should keep in mind this fact for the whole item.

Let's focus at the results per route-collector. For keeping a good extension in the thesis, we just show three of these route-collectors graphs. Then, we will analyze deeply the results per IXP, which are much more interesting from our point of view.

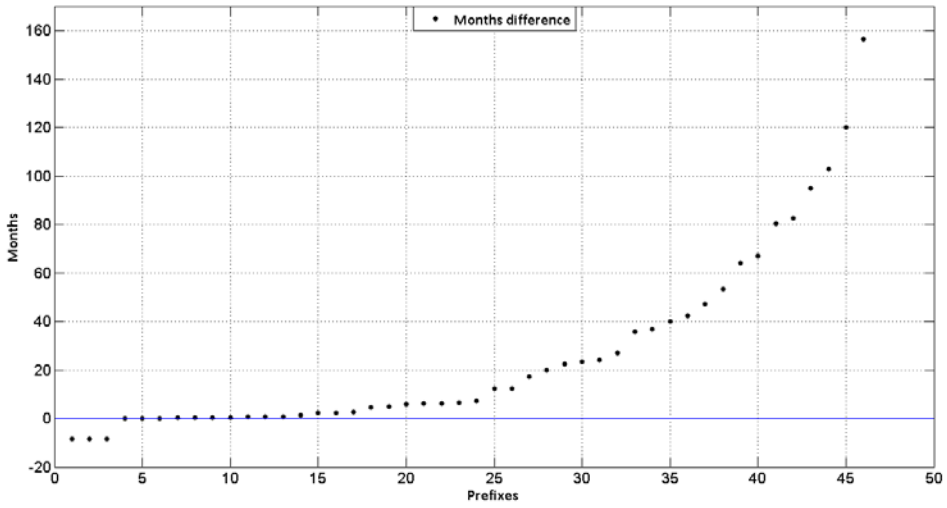


Figure 28: Months difference between the First Allocation time and the First time appearance of prefixes over time at route-collector: kixp.woodynet.pch.net (corresponding IXP: KIXP).

At this box, just 46 prefixes are announced, which represent the 4% of the previous graphs. In comparison to the 22.66% representation of the total number of prefixes announced at the corresponding IXP, as we will see later.

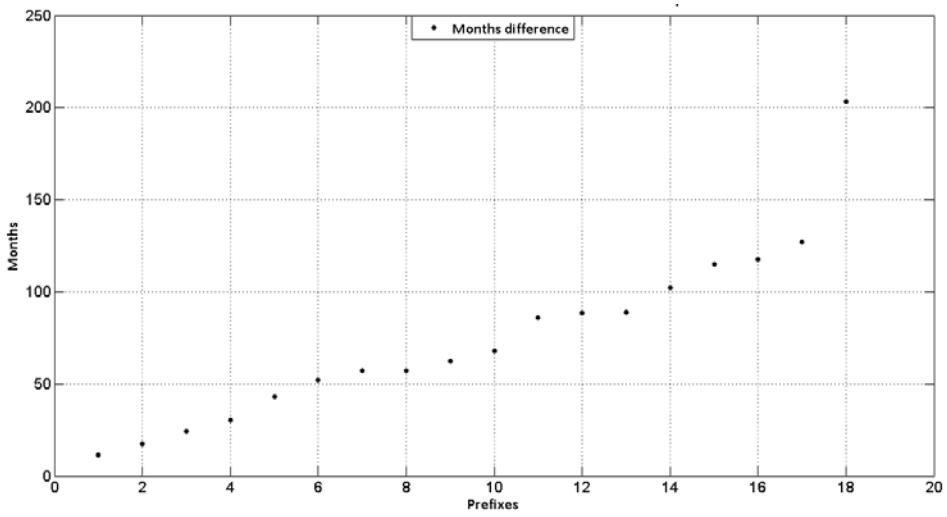


Figure 29: Months difference between the First Allocation time and the First time appearance of prefixes over time at route-collector: route-collector.krt.pch.net (corresponding IXP: SIxP).

In this case, this route-collector announces all the prefixes that appear in the corresponding IXP, which is SIxP. As we can see, the prefixes appear after their allocation date since the months difference is always positive.

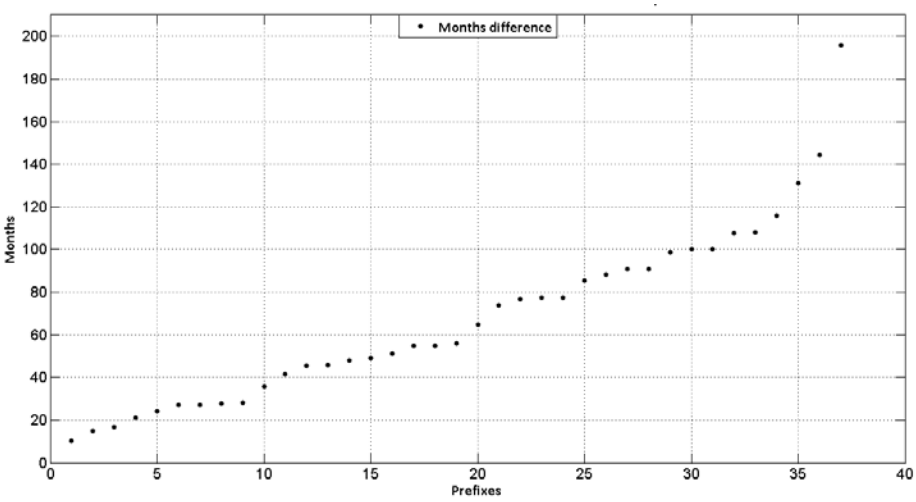


Figure 30: Months difference between the First Allocation time and the First time appearance of prefixes over time at route-collector: route-collector.los.pch.net (corresponding IXP: NIXP).

In Nigeria, we got that this route-collector announces huge months differences in almost all its prefixes from their allocation date. Besides, we are also able to check that those prefixes were not announced before their allocation date. Finally, in this box, we see that this is the route-collector which announced the 100% of prefixes from the IXP, since the IXP in Ibadan is no longer operational.

From these graphs we see that some of the prefixes were announced after the allocation date , whereas most of them were used long time after the allocation date. Moreover, we can detect which route-collector announces more prefixes at each IXP, which is interesting from the point of view of the ISPs.

We also computed another three graphs for the same IXPs containing the months difference between the last allocation date of a prefix and its first time appearance below.

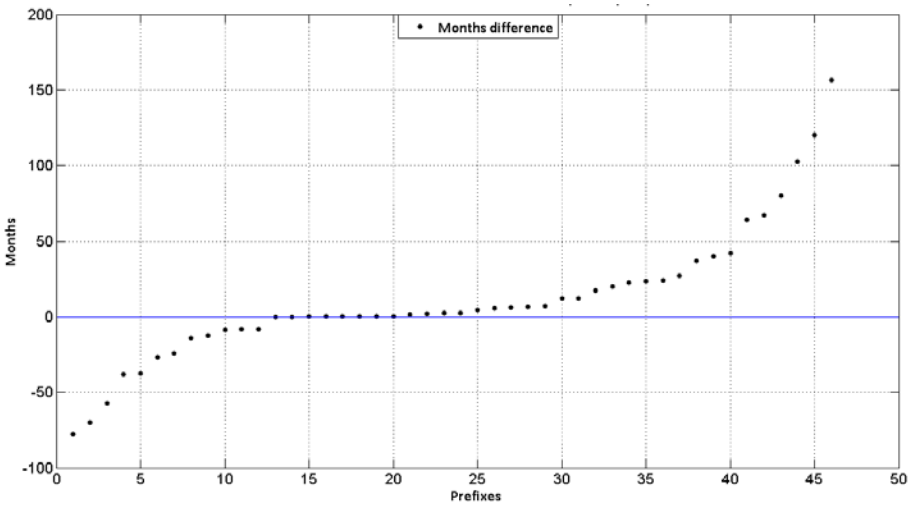


Figure 31: Months difference between the Last Allocation time and the First time appearance of prefixes over time at route-collector: kixp.woodynet.pch.net (corresponding IXP: KIXP).

At this box, 12 prefixes are announced before their last allocation due to the fact they appeared most probably when they were allocated for the first time by other operator. Then, we have a similar graph from the last of these prefixes, with approximately the same months difference.

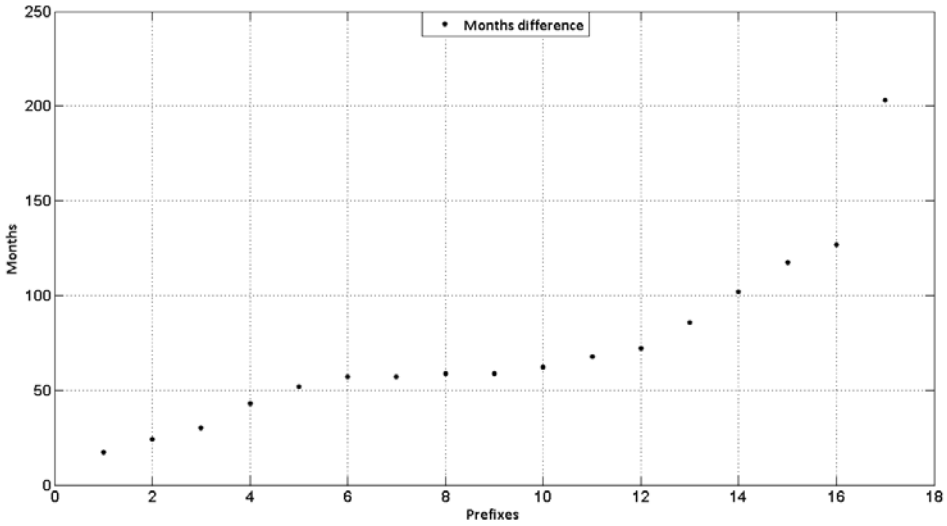


Figure 32: Months difference between the Last Allocation time and the First time appearance of prefixes over time at route-collector: route-collector.krt.pch.net (corresponding IXP: SIxP).

At SIxP, we detect that one of the prefixes is missing, which means that this prefix has not assigned another allocation date than the one considered as first allocation date. Despite this fact, the graph is pretty much the same as before.

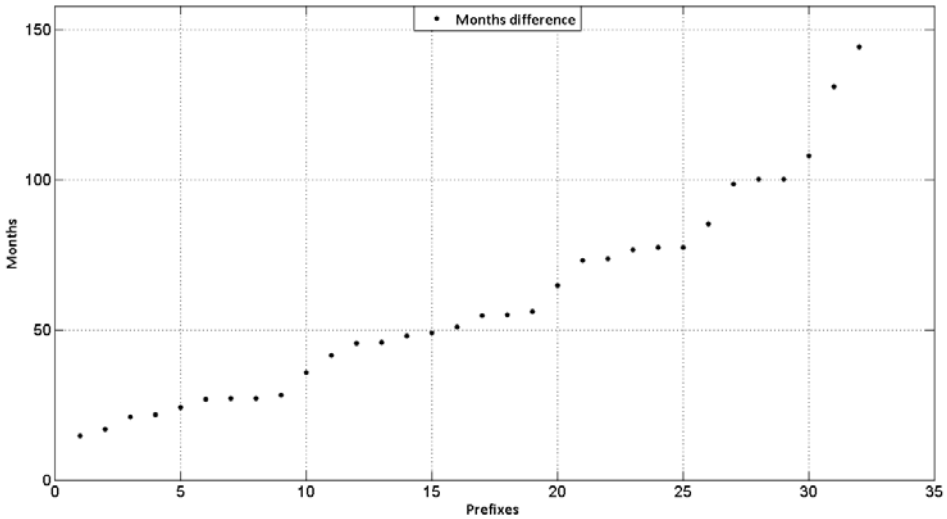


Figure 33: Months difference between the Last Allocation time and the First time appearance of prefixes over time at route-collector: route-collector.los.pch.net (corresponding IXP: NIXP).

As happened with SIxP, there are some prefixes with no last allocation date, and therefore they are not seen in this graph. In this box, just 5 prefixes are missing. However, the months difference is lower for the prefixes with biggest months differences. The evolution of the curve is pretty much equal to the graph that considers the first allocation date.

5.2.4 Results and graphs (part II)

In this section, we are going to focus first in the graphs that consider the months difference between the first allocation and the first appearance dates per IXP, and later the results that show the elapsed time in months between the last allocation and the first appearance dates.

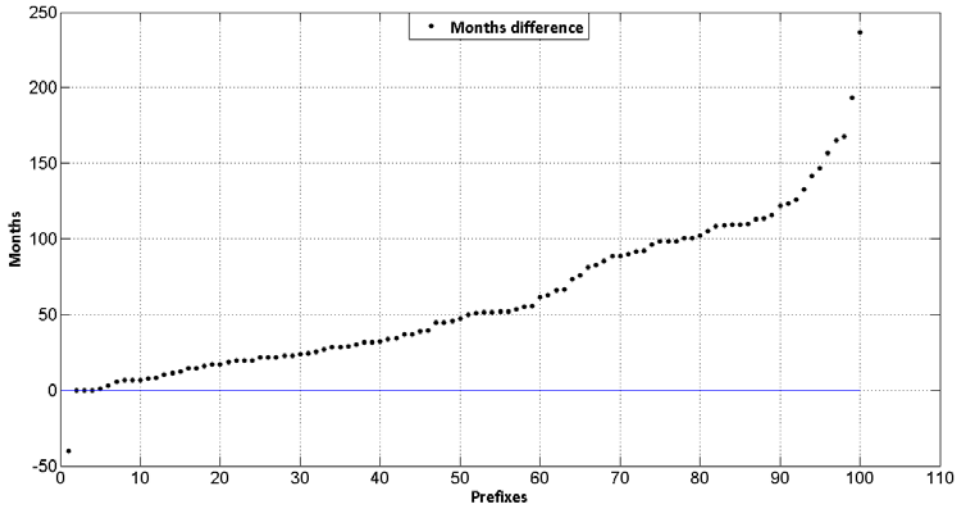


Figure 34: Months difference between the First Allocation time and the First time appearance of prefixes over time at CAIX (EG).

At CAIX, we found 8.71% prefixes of the total number of prefixes we seen at any IXP. The months difference evolution is linear till the prefixes that appear after 4 years since their allocation date, which are the 55% of the prefixes announced at this IXP. The first three prefixes are announced before their allocation date (due to the fact they appeared most probably when they were allocated for the first time by other operator) and next two prefixes appeared for the first time between the first and the second day after the allocation date.

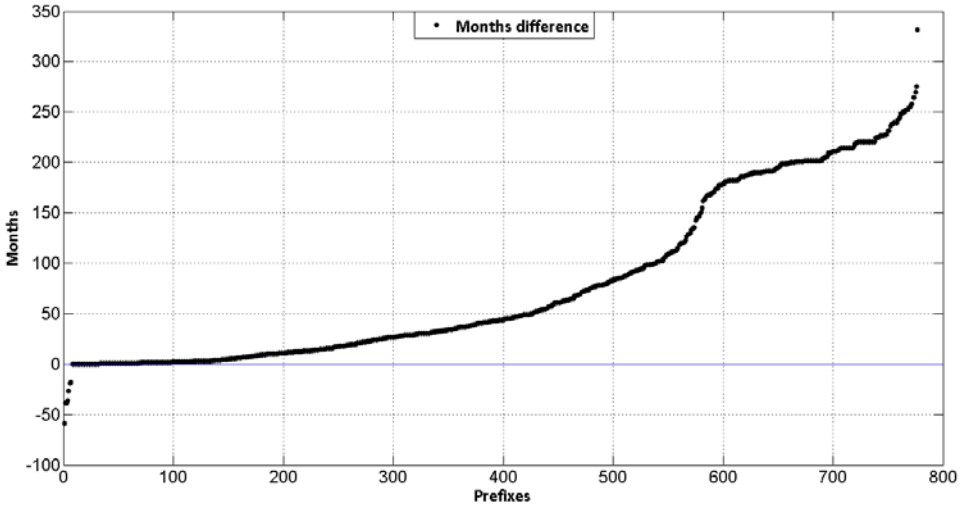


Figure 35: Months difference between the First Allocation time and the First time appearance of prefixes over time at CINX (ZA).

At CINX, 67.86% of the prefixes that were announced in the AFRINIC region at any IXP are seen. In addition, 7 prefixes are announced before their allocation date and just one is announced at the same day. We also see that 26.19% of the prefixes seen at this graph are announced at the same year of their allocation date.

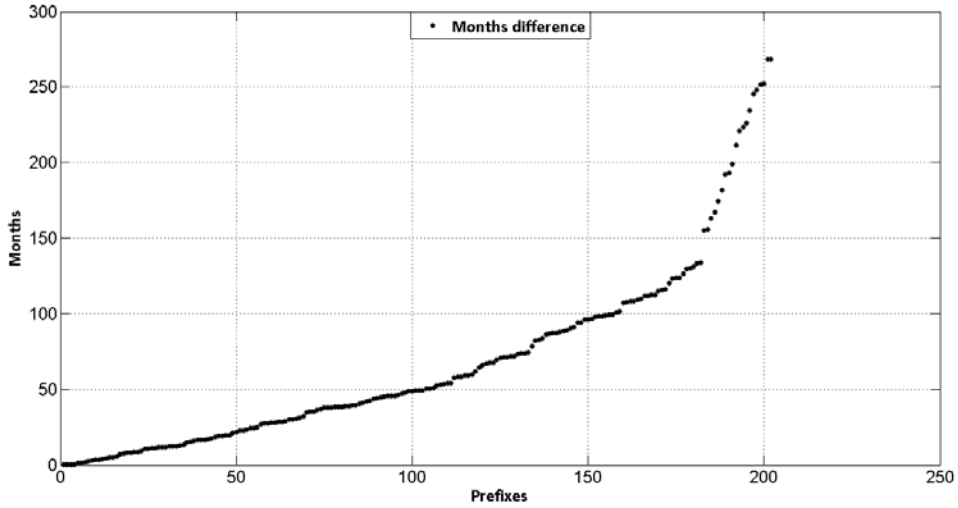


Figure 36: Months difference between the First Allocation time and the First time appearance of prefixes over time at DINX (ZA).

At DINX, 17.6% of the prefixes announced in the AFRINIC region at any IXP are seen. There are not prefixes announced before their allocation date and the evolution of the months difference follows almost a linear curve till the 175th prefix.

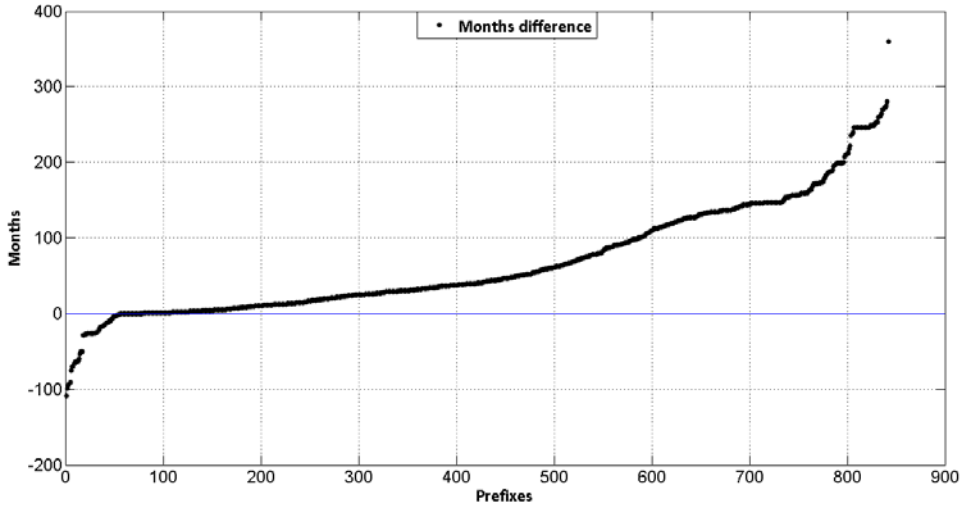


Figure 37: Months difference between the First Allocation time and the First time appearance of prefixes over time at JINX (ZA).

At JINX, we found 73.61% prefixes of the total number of prefixes we seen at any IXP. Since at CINX we got 67.86% of those prefixes, it means that some of the prefixes announced at this IXP are also at CINX.

There are 57 prefixes announced before their allocation date, which represent the 6.75% of the prefixes seen in the graph, and just one announced at the same day. Among all the prefixes of the graph, 19% of them are announced the same year of their allocation date.

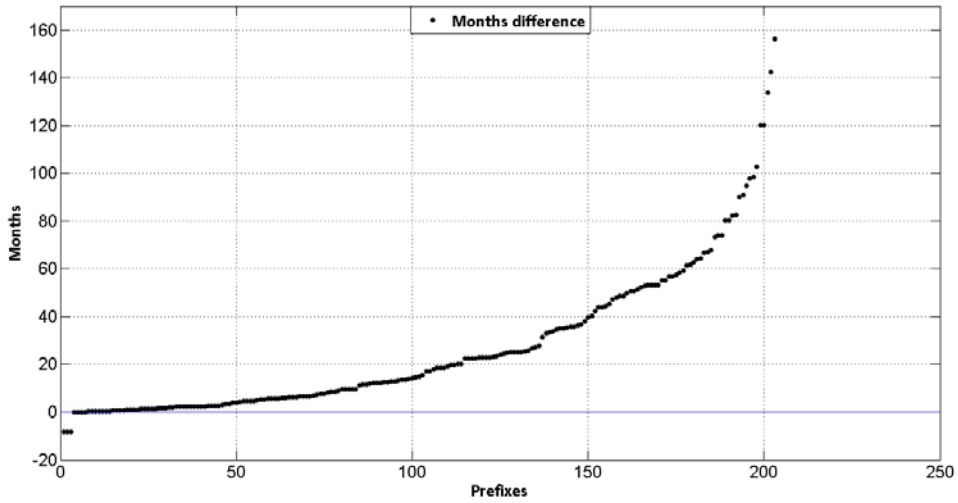


Figure 38: Months difference between the First Allocation time and the First time appearance of prefixes over time at KIXP (KE).

At KIXP, we found a clear exponential evolution of the months difference between the first allocation date and the first appearance. Just 3 prefixes are announced before the allocation date and the graph shows a 17.68% of the prefixes announced at any IXP.

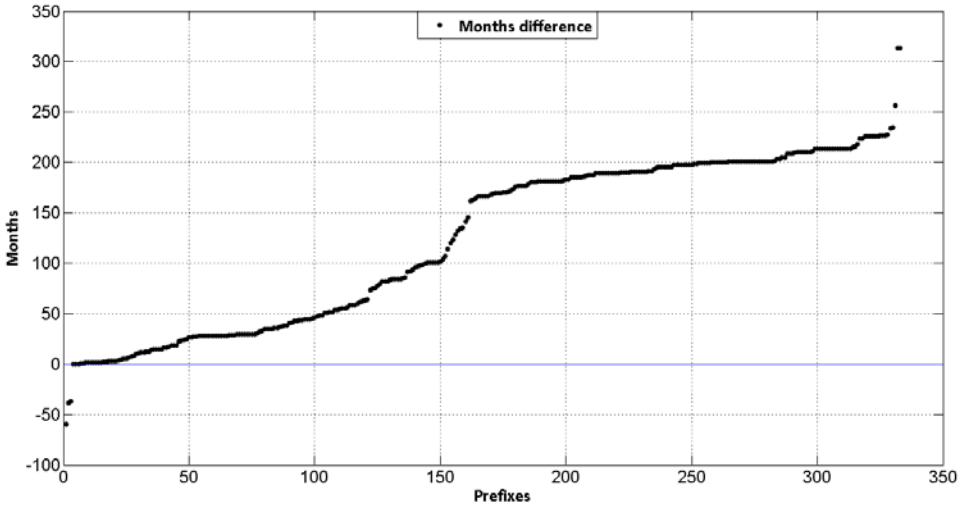


Figure 39: Months difference between the First Allocation time and the First time appearance of prefixes over time at MIX (MZ).

At MIX, just 3 prefixes are announced before the allocation date. We also notice 29% of the prefixes announced at any IXP. The evolution after the 150 first prefixes, is similar to the end of the evolution of CINX. This could be a result of the twice allocation of some prefixes due to new customers in the region.

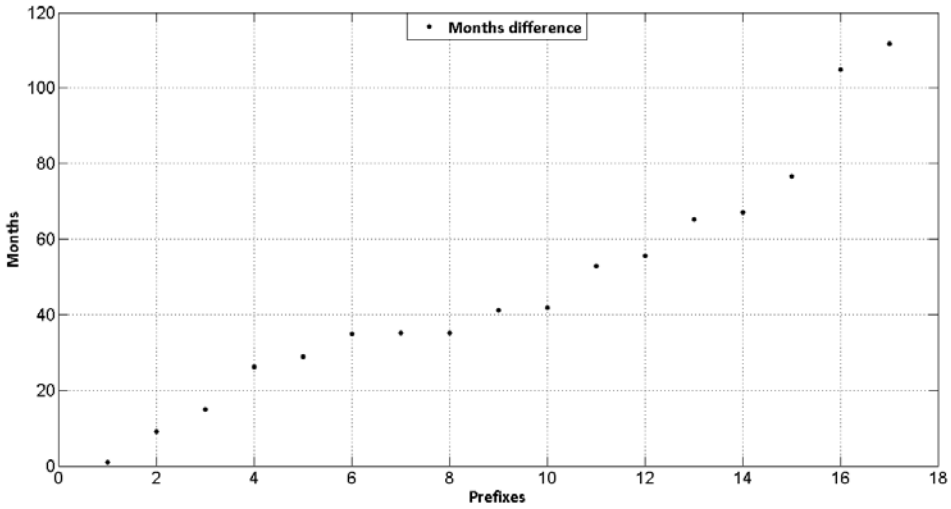


Figure 40: Months difference between the First Allocation time and the First time appearance of prefixes over time at MIXP (MW).

At MIXP, we have a small representation of the prefixes announced in the AFRINIC region by PCH, just 1.48% of the prefixes. It means that MIXP has less peering data collected in PCH than other IXPs.

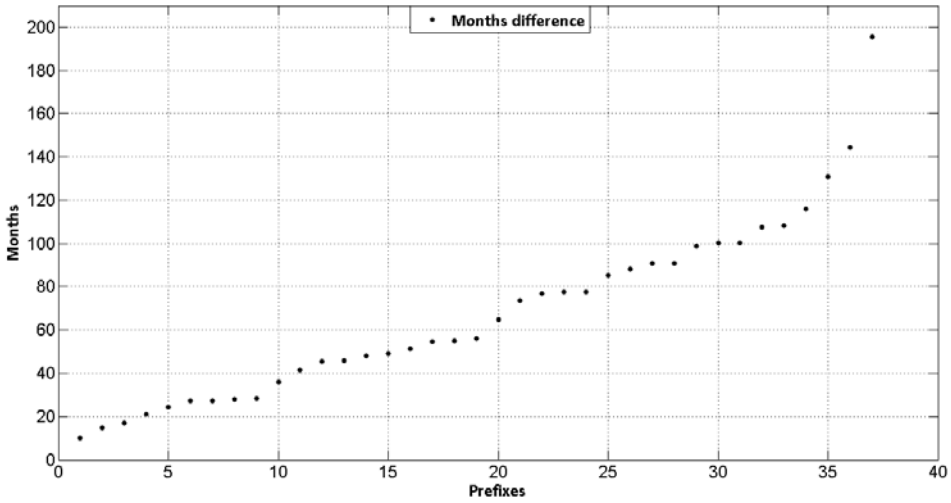


Figure 41: Months difference between the First Allocation time and the First time appearance of prefixes over time at NIXP (NG).

At NIXP, 3.22% of the prefixes announced in the AFRINIC region are in the peering data at PCH, and most of them are announced after a year of their allocation date.

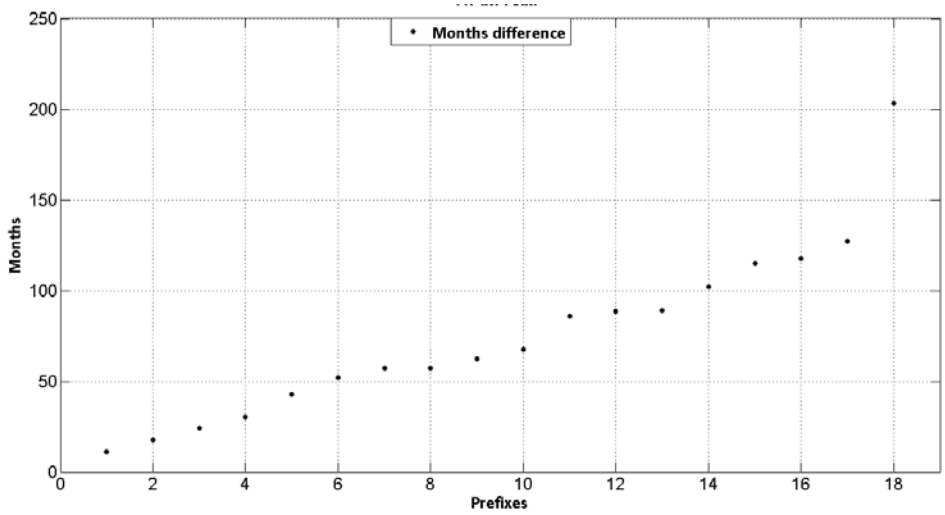


Figure 42: Months difference between the First Allocation time and the First time appearance of prefixes over time at SIxP (SD).

At SIxP, we have 1.57% of the prefixes announced in AFRINIC at any IXP. If we compare this graph with the one of the route-collector given before, we check that the most important box at this IXP is route-collector.krt.pch.net.

In conclusion, the IXPs which peer most with PCH are JINX and CINX. The ones less used for peering are SIxP and MIXP. Note that TunIXP has not a graph, because of the fact that its prefixes have not valid allocation dates in our databases.

Let's examine now the results that consider the months difference between the last allocation and the first appearance dates.

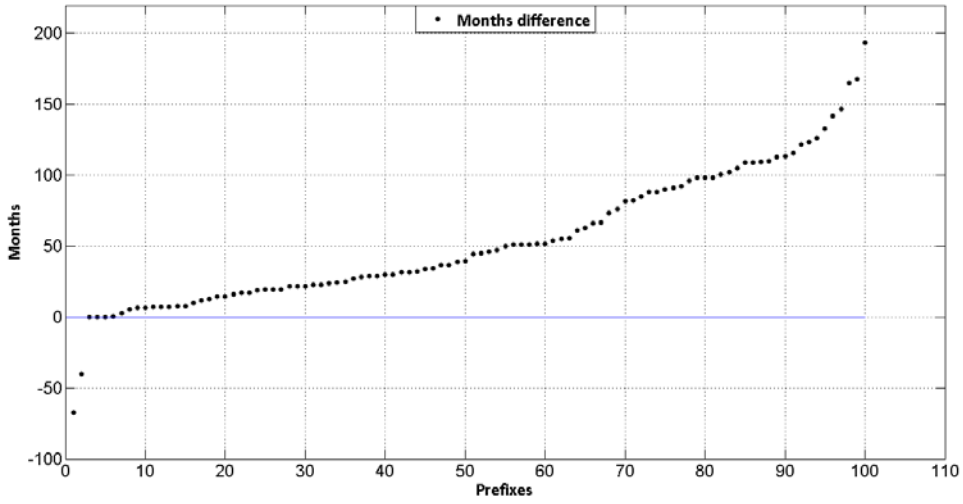


Figure 43: Months difference between the Last Allocation time and the First time appearance of prefixes over time at CAIX (EG).

At CAIX, 4 prefixes are announced before their last allocation date, just one more than in the graph that measures the months difference between the first allocation date of a prefix and its first appearance date. The evolution of the graph is pretty equal than before, but with smaller ranges between the months differences.

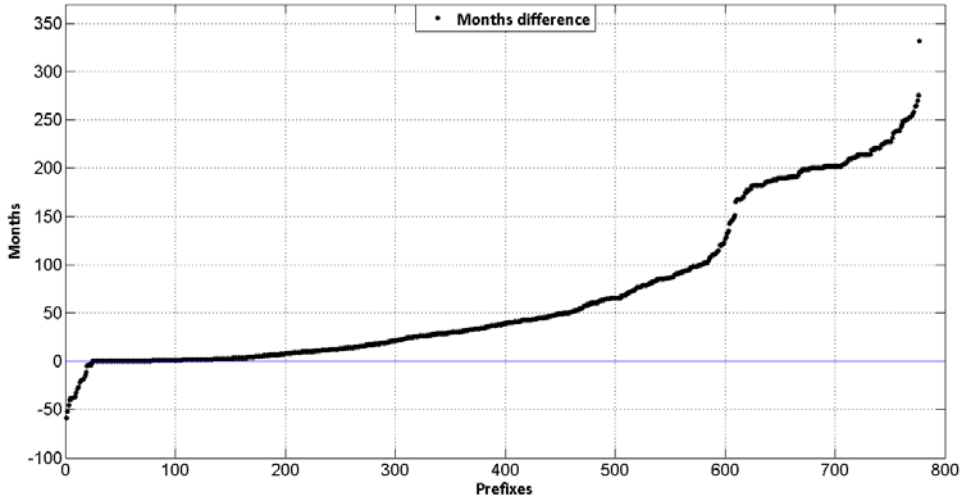


Figure 44: Months difference between the Last Allocation time and the First time appearance of prefixes over time at CINX (ZA).

At CINX, 24 prefixes are announced before their last allocation date, 17 prefixes more than before. It is a clear effect of the twice allocation issue. Nevertheless that case, the evolution shown in this graph is almost the same than before.

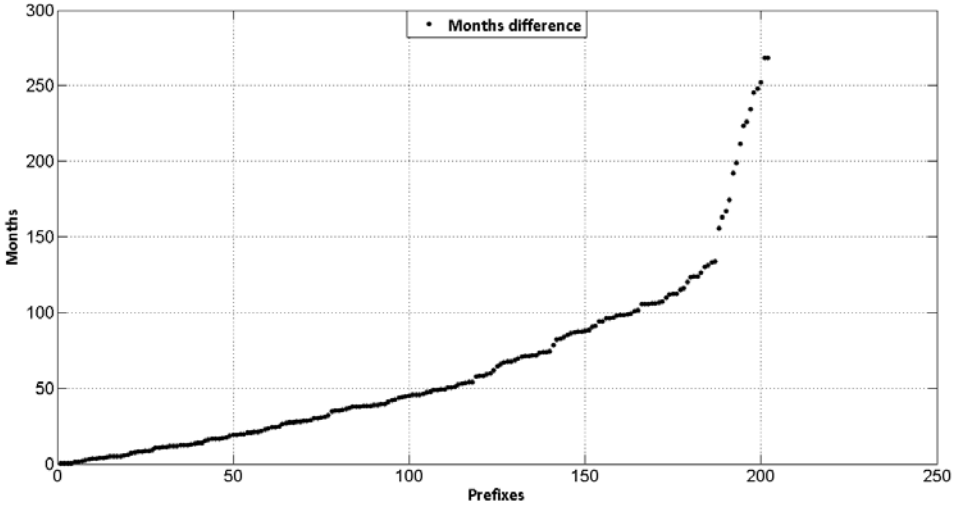


Figure 45: Months difference between the Last Allocation time and the First time appearance of prefixes over time at DINX (ZA).

At DINX, we do not see a difference with the graph that considers the first allocation date and the last allocation date.

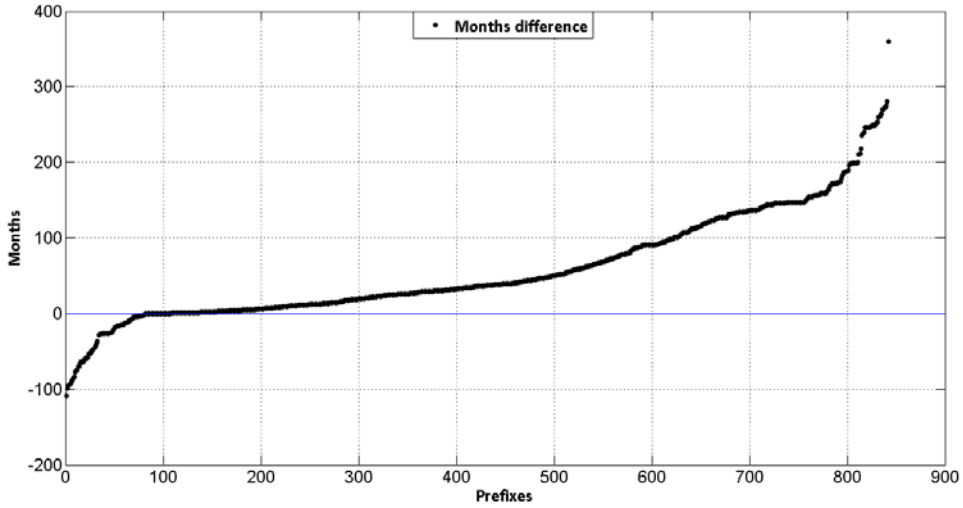


Figure 46: Months difference between the Last Allocation time and the First time appearance of prefixes over time at JINX (ZA).

At JINX, 84 prefixes are announced before their last allocation date, 27 more networks than before. But, the graph from these prefixes is equal to the previous one at JINX.

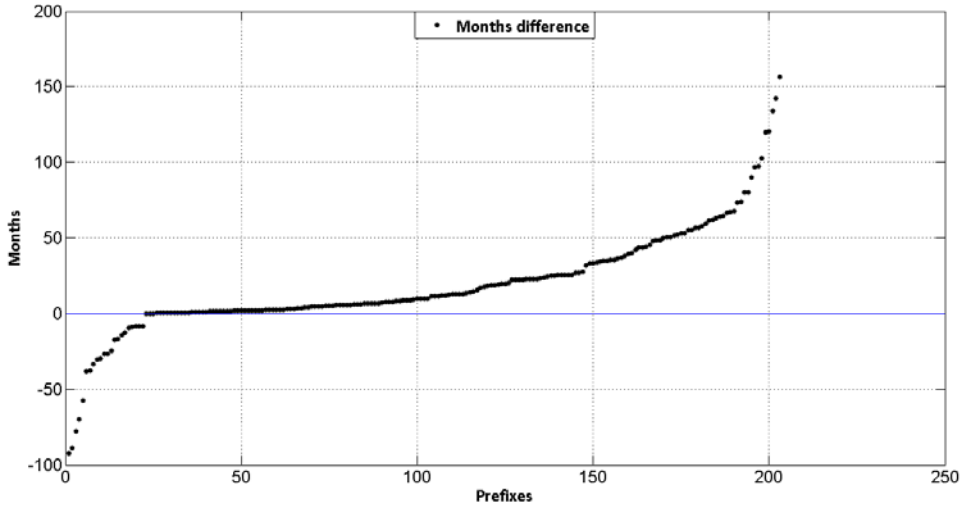


Figure 47: Months difference between the Last Allocation time and the First time appearance of prefixes over time at KIXP (KE).

At KIXP, we see that 22 networks are announced before their last allocation date, 19 more prefixes than before. However, the months difference for the rest of the prefixes announced is pretty much the same than before.

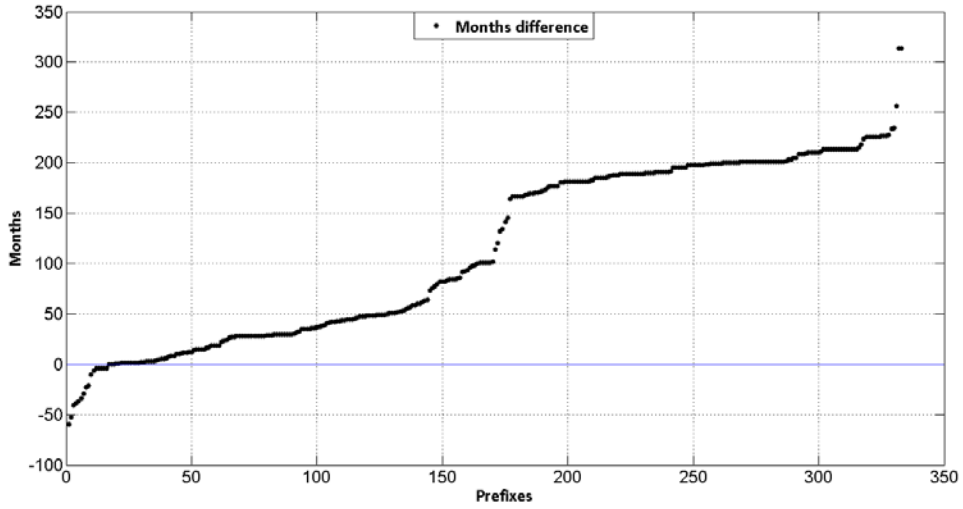


Figure 48: Months difference between the Last Allocation time and the First time appearance of prefixes over time at MIX (MZ).

At MIX, we notice that 13 more prefixes are announced before their allocation date than before. But the months difference for the rest of them in the graph is almost equal to the one before.

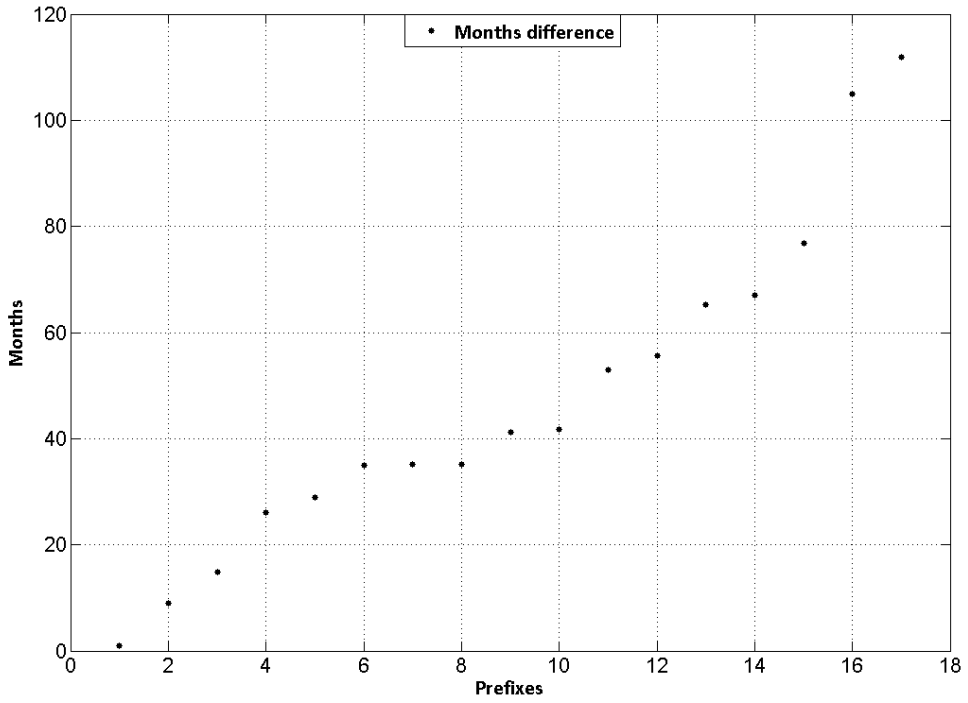


Figure 49: Months difference between the Last Allocation time and the First time appearance of prefixes over time at MIXP (MW).

At MIXP, we see no differences between the graphs.

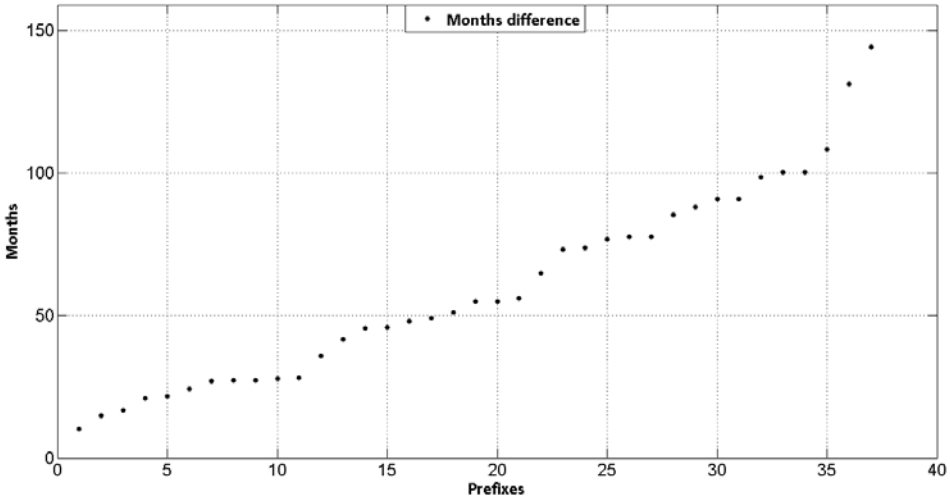


Figure 50: Months difference between the Last Allocation time and the First time appearance of prefixes over time at NIXP (NG).

As well as in MIXP, in NIXP we have no difference between the graph that consider the first allocation date and the last allocation date.

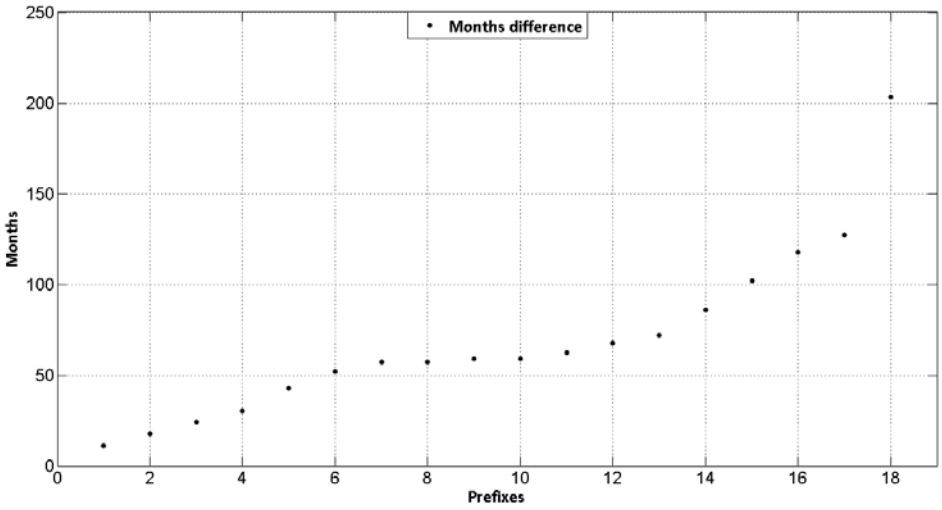


Figure 51: Months difference between the Last Allocation time and the First time appearance of prefixes over time at SIxP (SD).

As well as in the previous two IXPs, in SIxP we have no difference between the graph that consider the first allocation date and the last allocation date.

To sum up, the IXPs mostly used for peering in PCH (JINX and CINX), have more reallocated prefixes than the rest of the IXPs. Moreover, the ones less used did not show any change in their graphs.

5.3 Number and list of different prefixes visible in the data collected by PCH route-collectors deployed at an African IXP

The aim of this experiment is to let Regional (AFRINIC) and National Registries know whether organizations to which they allocate prefixes actually announce them in PCH.

The experiment is going to be explained in several subparts. The first subpart will be the algorithm's explanation that returns the visible prefixes collected in PCH boxes, the second one will show the extra resources needed to create the graph and the third one will contemplate the results.

5.3.1 Algorithm

Once more, we are going to describe the algorithm with a flowchart (see figure 52 in the next page). For this task, first we create a folder where we will store the different prefixes of an IXP over time in a file. We have to take also into account that if that folder is already created, we should remove it in order not to append the information to a file that was previously created and we create it again.

Once we have successfully created the folder, we need a list where all the years are defined, in order to traverse all the information in the database. In addition, we define the continent we want to study and we extract all the IXPs with their corresponding route-collectors from the table where we had classified them. Afterwards, we create a dictionary (*IXP_collectors*) with the IXPs as keys and the PCH boxes as values.

Then, we are going to use a loop, where we will select an IXP from the dictionary at each iteration. For each IXP, we create two dictionaries: one with the prefixes as keys and the years where they are announced as values, and another one with years as keys and the prefixes announced at that year as values. The reason why we create two dictionaries is to detect errors easily with them since we would be able to compare them, and fortunately, we will also use the dictionary with the years as keys in a future experiment.

After that, since some continents have no information for year, we should look for the existence of the specific continent and year. If we successfully find it, we add the year as a key in the dictionary *year_prefix*, and an empty list as a value.

At this step, we can select the desired information from our database, whose query in SQL is: select distinct *prefix*, *collector*, *country_code* from *continent_BGPdata_year* where *location* = *given_collector*. Hence, we will write a file per IXP containing the following data structure: prefix, corresponding route-collector, assigned CC of the route-collector. It is not needed to keep all these

data for the study itself, but it is helpful for detecting errors if some collector did not have correctly assigned the CC.

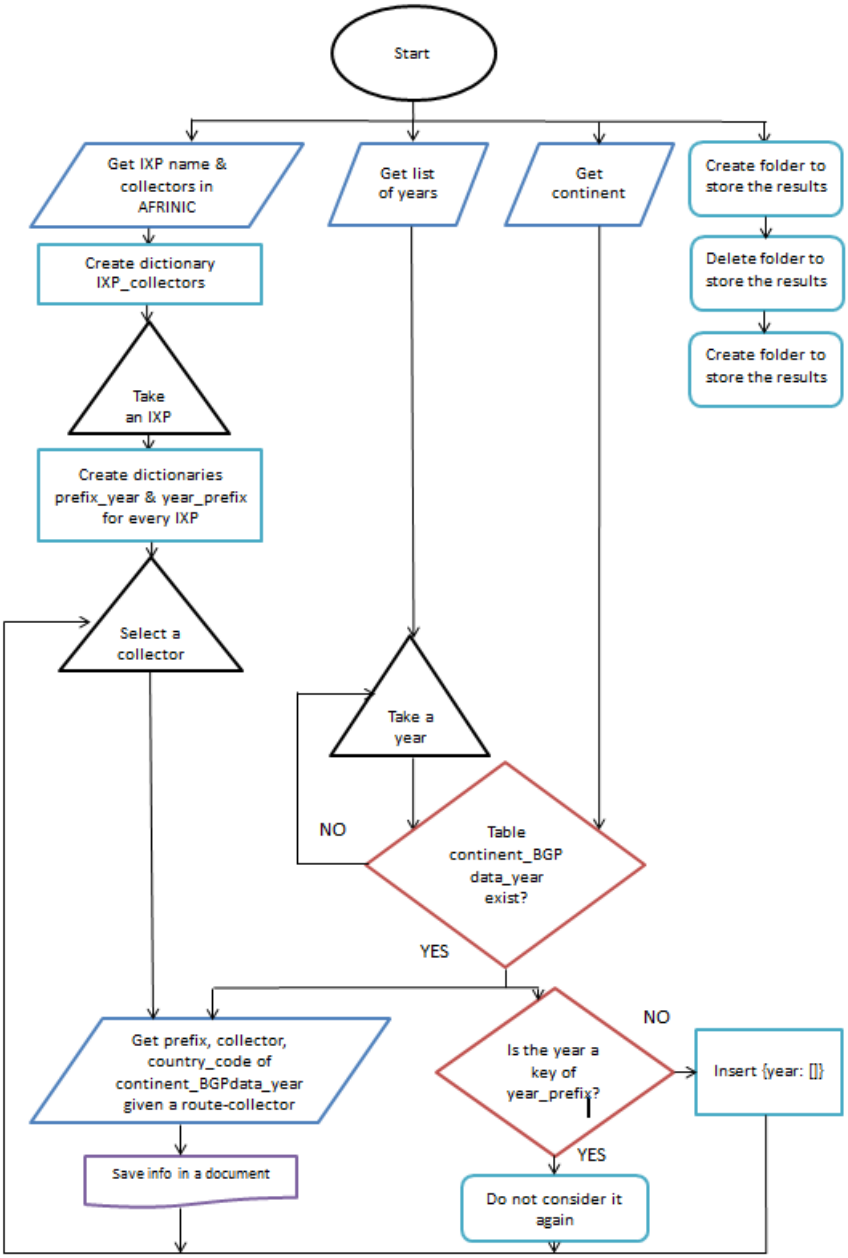


Figure 52: Descriptive flowchart with the input data formats and storage in memory.

Since we are returning the information corresponding to an IXP in a given continent and year, it may happen that some of the boxes won't have information at a concrete year, for instance if they were not peering yet. Thus, we check if we have returned any information. We should do this checking in order to fill correctly the dictionary *prefix_year*. Conversely, there will not be any possible error filling with the dictionary *year_prefix*, because if there is no information on a year, the empty list will remain the same.

After ensuring that we have data for an IXP in a continent, we also check if the prefix returned was already in the keys of the dictionary *prefix_year*. If not, we insert it and we create a list with the years in which it is announced. Furthermore, we append the prefix to the file where we are collecting the networks of an IXP, as well as its route-collector and its CC. Similarly, if the year studied is not in the keys of the dictionary *year_prefix*, we insert the year as a key and the prefix as an element of the list which contains the different prefixes already stored at the year.

Finally, in order to know the number of prefixes stored in each file, we can count the lines in the file in a terminal using the command "wc -l <name of the created file>" or printing the length of the dictionary. We could also have extracted the data from the database without specifying the box we wanted to check, but since we want to store the prefixes associated to an IXP over time, in regions where we have plenty route-collectors, we will be spending more time checking if the data corresponds to any box of the IXP under study. Our approach makes more connections to the database, but we save at least a couple of verifications per IXP.

5.3.2 Extra resources

We reused the code described in 5.3.1 to extract the number of prefixes visible at each box in PCH. Instead of creating two dictionaries per IXP, we created them per route-collector. Then, for extracting this information, we will have as outputs a file per route-collector and we will know how many prefixes are announced at each route-collector either printing the keys length of the dictionary prefix_year or using the command "wc -l <name of the created file>".

5.3.3 Results

Firstly, we are going to show the number of prefixes that are announced at PCH in each box.

CC	IXP	Route-collector	Number of visible prefixes
EG	CAIX	route-collector.cai.pch.net	6767
ZA	JINX	jinx.woodynet.pch.net	5557
ZA	CINX	route-collector.cpt.pch.net	5412
ZA	JINX	route-collector.jnb.pch.net	4826
MZ	MIX	route-collector.mpm.pch.net	3323
KE	KIXP	route-collector.nbo.pch.net	3077
ZA	JINX	router.jnb.woodynet.net	1846
ZA	DINX	route-collector.dur.pch.net	1816
KE	KIXP	kixp.woodynet.pch.net	1540
NG	NIXP	route-collector.los.pch.net	1140
SD	SIxP	route-collector.krt.pch.net	496
MW	MIXP	route-collector.blz.pch.net	107
TN	TN	route-collector.tun.pch.net	18

Table 17: Number of distinct prefixes visible per PCH box.

From these results, we extract that the top three route-collectors most used for peering according to PCH dataset are route-collector.cai.pch.net, jinx.woodynet.pch.net and route-collector.cpt.pch.net. Regarding the top three route-collectors less used for peering are route-collector.tun.pch.net, route-collector.blz.pch.net and route-collector.krt.pch.net.

Therefore, it seems that the IXPs that collect most peering data are CAIX (located in Egypt), JINX and CINX (located in South Africa). Similarly, the IXPs that should be collecting less peering information are TunIXP, MIXP and SIxP, which coincides with the IXPs that had not reallocated prefixes and had not changes in their graphs, probably because of their date of launch.

Secondly, we show the number of prefixes visible at each IXP according to PCH dataset.

CC	IXP	Number of visible prefixes
ZA	JINX	9690
EG	CAIX	6767
ZA	CINX	5412
KE	KIXP	4235
MZ	MIX	3323
ZA	DINX	1816
NG	NIXP	1140
SD	SIxP	496
MW	MIXP	107
TN	TunIXP	18

Table 18: Number of distinct prefixes visible per IXP.

We observe that the most used IXPs are JINX, CAIX and CINX, as we deduced thanks to the previous table. CAIX was not the second IXP with so many prefixes announced in the previous graph (months difference between allocation date and appearance dates), but it is the second IXP with the biggest amount of prefixes visible. Similarly, the IXPs that collect less peering information according to PCH are TunIXP, MIXP and SIxP.

5.4 Number and list of ASes visible in the data collected by PCH route-collectors deployed at an African IXP

This item is going to be explained in three subparts. The first part will be the algorithm's explanation that returns the visible Autonomous Systems (ASes) collected in PCH boxes, the second one will show the extra resources needed to accomplish the graph and the third one will contemplate the results.

The goal of this experiment is to know whether AS allocation by AFRINIC and national registries are used and visible in PCH data.

5.4.1 Algorithm

This script is pretty similar to the one in the previous section, despite of the query and how we extract the ASes from our data. Therefore, there will not be a flowchart in this section.

We fetch from the database the AS path list and we will consider both the origin and all the ASes in the path. With the returned data, we split the complete path (that is separated by spaces) and we take the last item of the list, which corresponds to the origin AS. It is important to check if the AS is valid, in order to correctly insert the associated parameters into the file with the following structure: AS number, corresponding collector, CC of the collector assigned to the AS.

It is interesting to remark that in some of the downloaded files, the structure where several ASes were collected as origin ones, was: {*origin_AS1, originAS_2*}. Thus, in such case, we had to detect the change in the data format, deleting the curly braces and splitting the data by comma.

In order to be completely sure that there will not be any error in the document, we create an independent list per IXP where we will store the origin ASes seen, even if they are not valid, in order not to insert them neither in the dictionary with all the ASes nor in the file per IXP. Every time that a valid prefix is not in the list, we add it and we also append it to the document.

We create another file where we collect all the ASes seen at each IXP. For this task, we use the dictionary that store the ASes as keys and the years where they are announced as values.

As a final note, we could have extracted the data from the database without specifying the route-collector we wanted to check, but since we want to store the ASes associated to an IXP over time, in regions where we have lots of boxes, we would spend too much time checking whether the data corresponds to any box of the IXP under study. As was previously stated, our approach makes more connections to the database, but we save at least a couple of verifications per IXP. Actually, we could also have done two scripts: one for the origin ASes and another one for collecting all of them. Although in such case, we will have to do the same connections to the database twice, since we create the list of origin ASes and the dictionary with all the ASes at the same time.

5.4.2 Extra resources

We used the code described in 5.4.1 to extract the number of ASes visible at each box in PCH. Instead of creating two dictionaries per IXP, we created them per route-collector, as well as the independent list for the origin ASes. Then, for extracting this information, we have as an output

a file per route-collector and we know how many ASes are announced at each route-collector either printing the keys length of the dictionary with the ASes as keys or using the same command as before.

5.4.3 Results

As we did in the previous section, we start showing the number of ASes that are announced at PCH in each box, but just for the origin ASNs (AS Numbers).

CC	IXP	Route-collector	Number of Visible Origin ASNs
ZA	CINX	route-collector.cpt.pch.net	482
ZA	JINX	route-collector.jnb.pch.net	450
KE	KIXP	kixp.woodynet.pch.net	229
KE	KIXP	route-collector.nbo.pch.net	182
ZA	DINX	route-collector.dur.pch.net	174
ZA	JINX	jinx.woodynet.pch.net	149
MZ	MIX	route-collector.mpm.pch.net	129
EG	CAIX	route-collector.cai.pch.net	70
NG	NIXP	route-collector.los.pch.net	69
ZA	JINX	router.jnb.woodynet.net	49
MW	MIXP	route-collector.blz.pch.net	13
SD	SIxP	route-collector.krt.pch.net	9
TN	TunIXP	route-collector.tun.pch.net	2

Table 19: Number of distinct Origin ASNs visible per PCH box.

For our purposes, it was interesting to have the number of origin ASes visible at every route-collector, in order to have a hint of which IXPs will be registering most peering data at PCH, which should be according to this table: CINX, JINX and KIXP. Besides, the top three IPXs with less peering data are TunIXP, SIxP and MIXP.

To change the subject of this table, we consider now both the origin ASes and all the ASes that are visible at an IXP (cf. table 20).

CC	IXP	Number of distinct visible Origin ASNs
ZA	JINX	515
ZA	CINX	482
KE	KIXP	362
ZA	DINX	175
MZ	MIX	129
RG	CAIX	70
NG	NIXP	69
MW	MIXP	13
SD	SIxP	9
TN	TunIXP	2

Table 20: Number of distinct Origin ASNs visible per IXP.

We can easily check with the previous table that the ASes visible at CINX are given by just one route-collector, some of the networks given at KIXP route-collectors are repeated, and therefore they are not considered twice. The same happens with JINX.

As expected, according to the origin ASes visible at an IXP, we have the same conclusion than the obtained per route-collector. The top three IXPs with most peering data at PCH are CINX, JINX and KIXP. In addition, the top three IXPs with less peering information according to PCH are TunIXP, SIxP and MIXP, whose visible ASes are given just by one box each.

Let's compare these results with the ones considering all the ASes, regardless if they are origin ones or not.

CC	IXP	Number of distinct visible ASNs
ZA	JINX	541
KE	KIXP	540
ZA	CINX	500
ZA	DINX	177
MZ	MIX	132
EG	CAIX	74
NG	NIXP	69
MW	MIXP	14
SD	SIxP	9
TN	TunIXP	2

Table 21: Number of distinct ASNs visible per IXP.

Despite there is not a huge difference between the origin ASNs visible at an IXP, we can see that KIXP has the biggest difference among all the IXPs. Besides, it is clear that all the ASes visible at the top three IXPs with less peering information are considering all the origin ASes found in each one of them.

Chapter 6: Statistics (part II)

This chapter details the studies developed for achieving the remaining thesis' objectives. For each one of them, we follow the same structure of the previous chapter. So, there is an introduction, a detailed description of the algorithms needed, an explanation of extra resources if needed, a graph or table with the results, and a brief discussion about them.

6.1. Prefix growth statistics per year at IXPs in PCH dataset

The goal of this experiment is to know whether prefix allocation by AFRINIC and national registries are used and visible per year in PCH dataset.

The experiment is going to be explained in several subparts. The first subpart will be the algorithm's explanation that returns the visible prefixes collected in PCH boxes per year, the second one will show the extra resources needed to accomplish the graph and the third one will show the results.

6.1.1 Algorithm

Given the script used in item 5.3, and taking into account that we created a dictionary per IXP with the years as keys and the prefixes announced per year as values, we can easily create a file with this dictionary called *year_prefix*. Once all the prefixes are extracted for an IXP, we ensure that we have not repeated prefixes (i.e. distinct prefixes) for each year converting the list into a set. Then we write a file per IXP with the distinct years, followed by a comma and the length of the set per line into the document.

Lastly, we could also have implemented this experiment with another script if the prefixes and the years were stored as a unique tuple of values in the database. We just reused the code because we want to save as many space in the server as possible.

6.1.2 Extra resources

We reused the code described in 6.1.1 to extract the number of prefixes visible at each box in PCH per year. Instead of creating two dictionaries per IXP, we created them per route-collector. Then, for extracting this information, we have as an output a file per route-collector and we know how many prefixes are announced at each route-collector per year with the same structure described before: year, number of prefixes at that year.

We can reuse the scripts created for MATLAB in the previous chapters, but changing the axes labels and the colors to print the values, as well as the legend per IXP and per route-collector.

6.1.3 Results and graphs

It is important to have in mind that not every ISP peer with PCH, although it is open peering policy, these graphs are biased. The graphs corresponding only to route-collectors are also biased since the RIR database dates were not updated until 2015, but just till 2012. For the remaining graphs in this item, the RIR database was updated. As in item 2, we are going to provide just 3 graphs for illustrating the route-collector information and then, we will analyze the evolution per IXP.

Let's focus on the same route-collectors than in item 2 of this chapter:

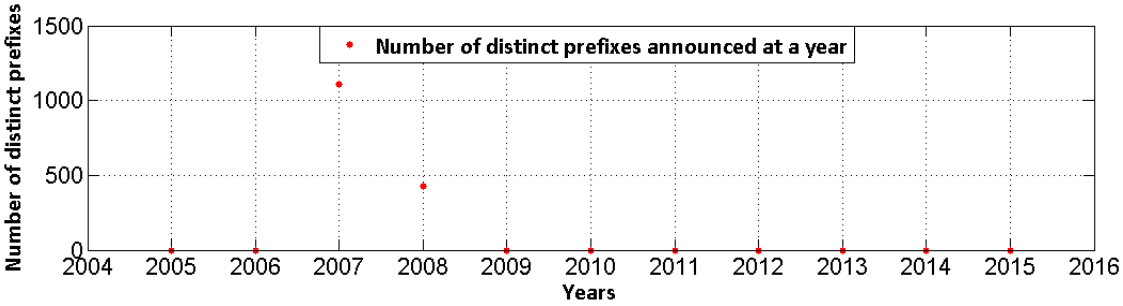


Figure 53: Evolution of distinct prefixes announced at route-collector *kixp* (KIXP).

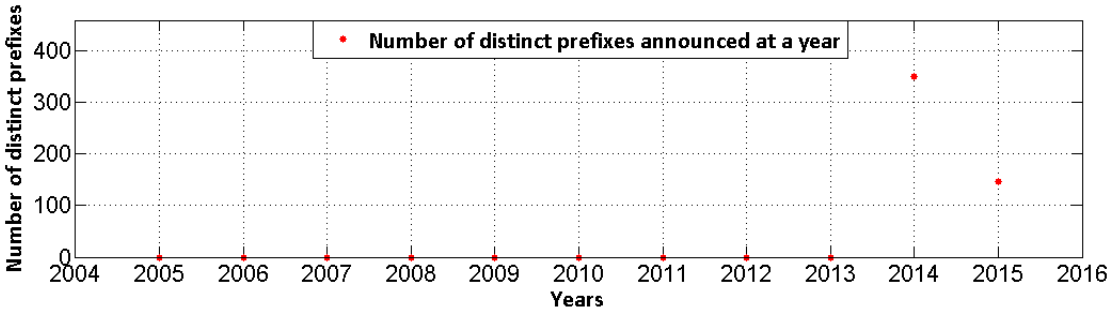


Figure 54: Evolution of distinct prefixes announced at route-collector *krt* (SIxP).

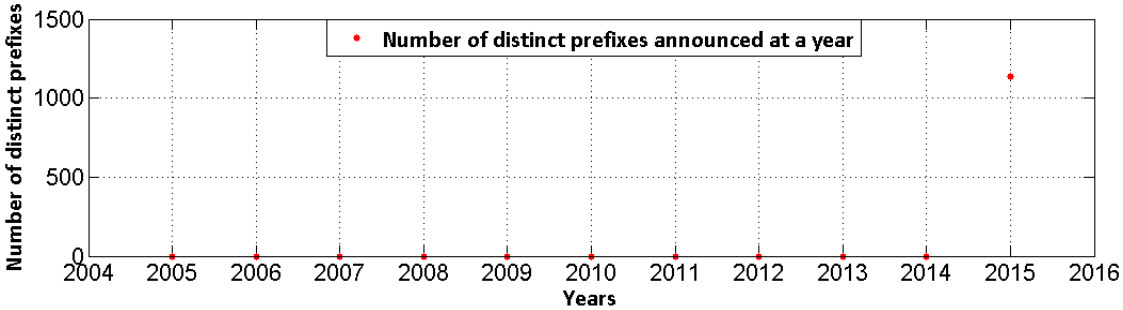


Figure 55: Evolution of distinct prefixes announced at route-collector *los* (NIXP).

From these results, we observe that the box of KIXP started to collect peering data in 2008 with a considerable amount of prefixes and that this amount was drastically reduced in 2009, the last year of peering data collected at PCH. In the case of SIxP, the box shows that has started

collecting data recently, from 2014 on. Finally, at NIXP, we see that the only box we have at that IXP has been storing peering information just in 2015.

Let's move to the IXPs results and their prefix growth.

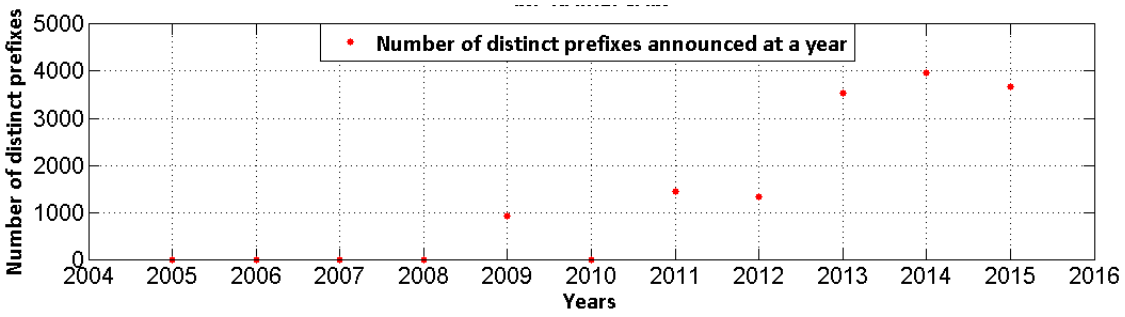


Figure 56: Evolution of distinct prefixes announced at CAIX (EG).

At CAIX, we have 2010 as a year in which there are not networks visible. It means that PCH has not collected data in those years. We notice that 13.74% prefixes are seen in 2009, regarding the distinct number of prefixes announced at that IXP over time; 20.47% for 2011 and 2012; and the ratio of announced prefixes between the last three years is 54.91%. Obviously, the amount of prefixes seen at the last year is a bit lower than in 2014 since the last year has not finished yet, and this will be the case for most of the IXPs.

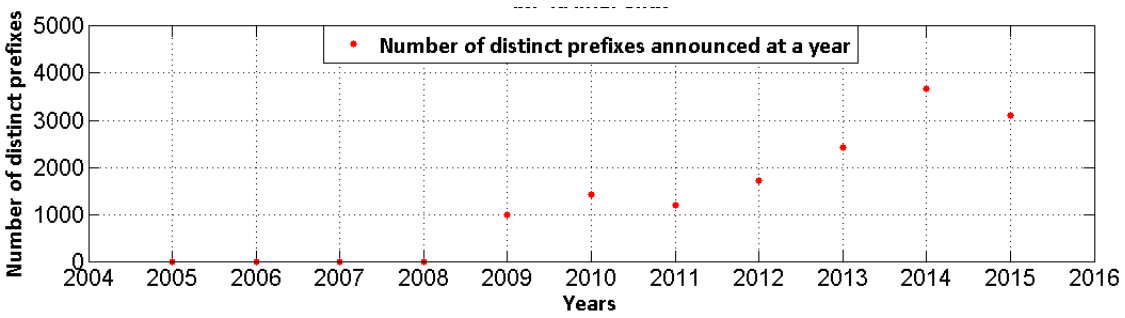


Figure 57: Evolution of distinct prefixes announced at CINX (ZA).

At CINX, we have a continuous evolution in the growth of the prefixes seen despite an inflection point in 2011. Therefore, the slope of the evolution is similar to CAIX. In 2009, 18.07% of the prefixes seen at CINX are visible; it increases to 26.39% in 2010 and it rises till 67.72% in 2014. As in CAIX, the year with most prefixes seen is 2014. We notice that it is interesting that the percentage of the first year has evolved to three times more till the year with most prefixes seen.

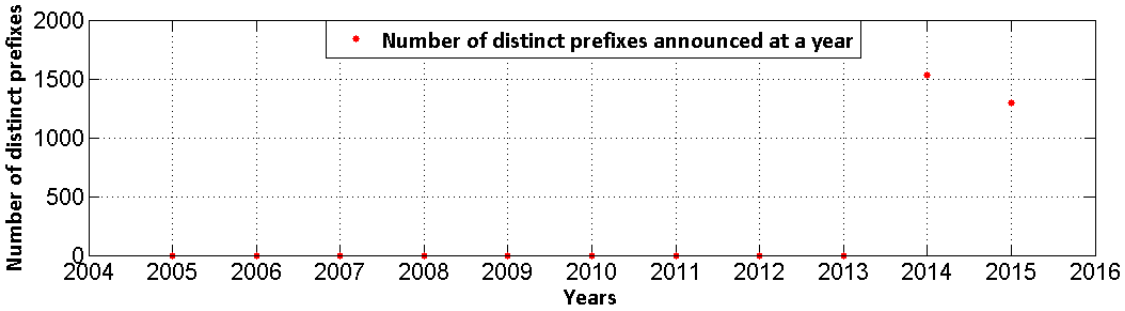


Figure 58: Evolution of distinct prefixes announced at DINX (ZA).

At DINX, 84.42% of the prefixes are seen in 2014 out of the total number of prefixes seen at the IXP, and 71.37% of the prefixes in 2015.

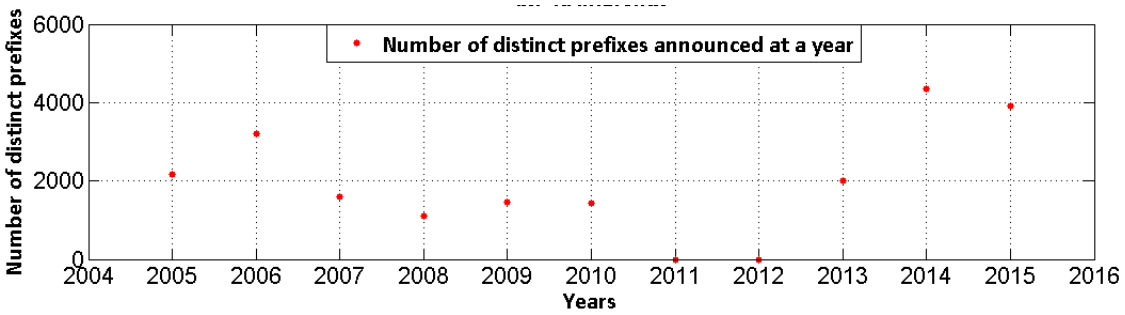


Figure 59: Evolution of distinct prefixes announced at JINX (ZA).

At JINX, we have two years in which there are not networks visible, since there was not collected PCH data in 2010 and 2011. Before this break, the first year with percentage of prefixes visible is 2005 with 22.29%, and after the break 2014 has 44.82%. If we compare this IXP with the case of CINX, we see that the range between percentages of the first year and the year with most prefixes seen is not that wide. CINX evolved three times the percentage in the first year, but JINX just doubled it.

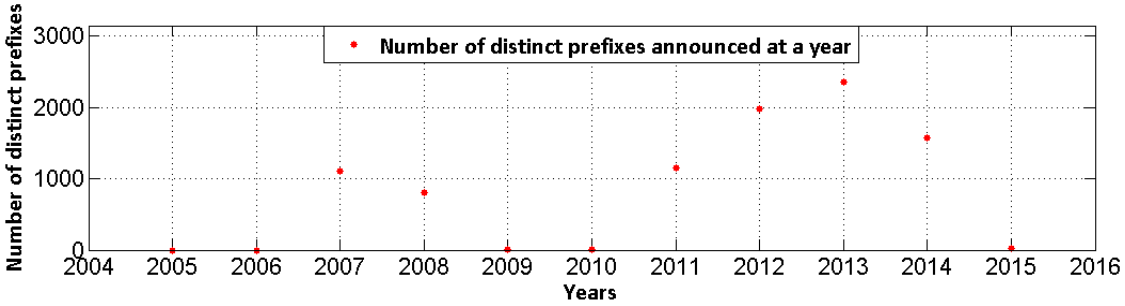


Figure 60: Evolution of distinct prefixes announced at KIXP (KE).

The first year in KIXP, 26.26% of the prefixes are seen of the total number of prefixes visible at the IXP. Then, after one year, there is a two year drop of prefixes seen at the IXP, to 8 and 12 prefixes respectively. In the previous IXPs, the drop was to 0 prefixes. Moreover, there is another drop in 2015 to 17 prefixes, because we have not collected all the information on 2015 yet. Maybe the data will be available at the end of the year.

Right after the drop, it continues evolving till 55.87% of the prefixes seen at the IXP in 2013, the year with the biggest amount of distinct prefixes seen at a year.

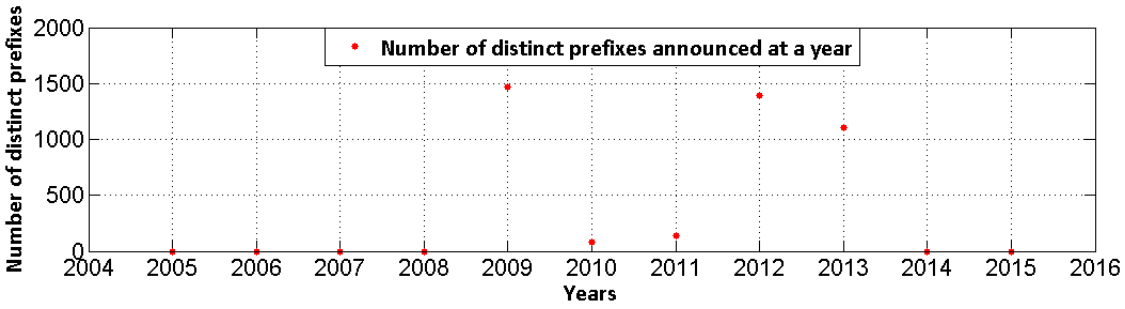


Figure 61: Evolution of distinct prefixes announced at MIX (MZ).

At MIX, the drop in the amount of prefixes seen in the IXP at a year is in 2010 and 2011, till around just 100 prefixes each. MIX case is special since the year with the biggest percentage of prefixes seen at a year is the first one, with a total of 44.27%. After the break, the percentage in 2011 is 41.83% and will be drastically reduced to 0% in 2014 till nowadays. It is reasonable to think that this IXP is not collecting PCH data anymore.

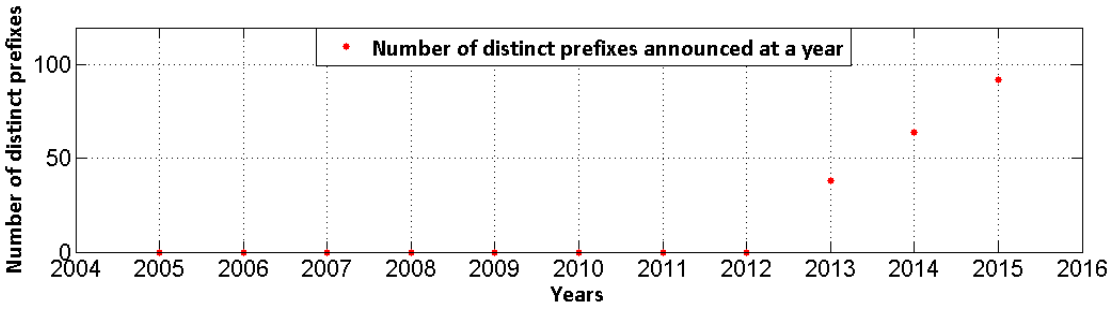


Figure 62: Evolution of distinct prefixes announced at MIXP (MW).

At MIXP, we are able to say that ISPs have not been peering with PCH till 2013. However, we found a high slope in the last years and that it is evolving from 35.51% in 2013 to 85.98% in 2015.

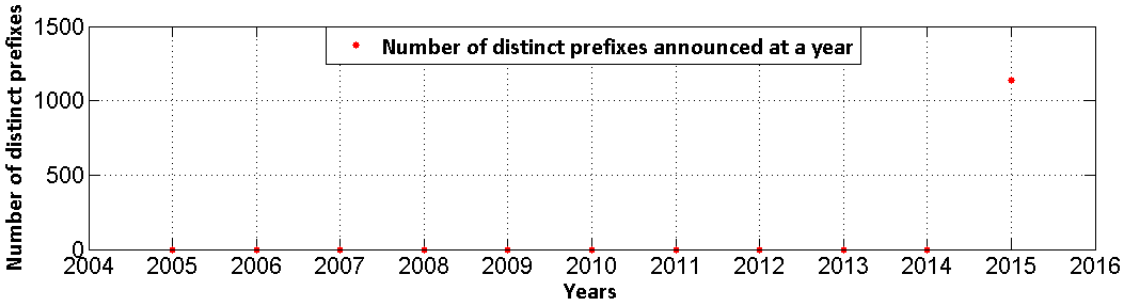


Figure 63: Evolution of distinct prefixes announced at NIXP (NG).

At NIXP, we have the 100% of distinct prefixes seen at the IXP in 2015. Hence, PCH is clearly starting to collect data at this IXP.

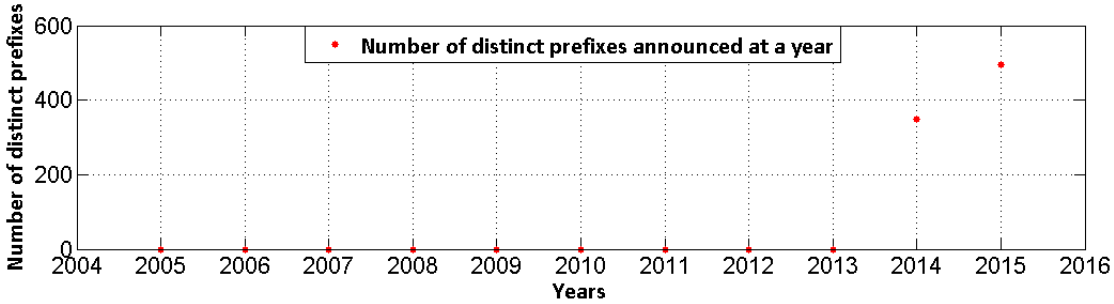


Figure 64: Evolution of distinct prefixes announced at SIxP (SD).

At SIxP, 70.56% of the prefixes visible at the IXP are seen in 2014 and all of them in 2015. Then, it is growing from 2014 on.

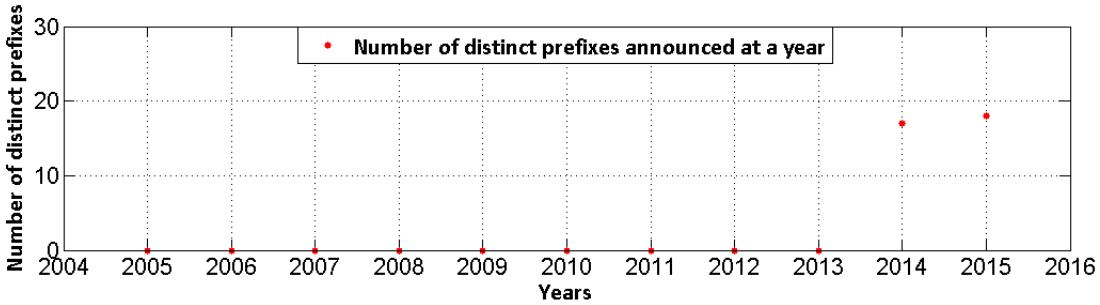


Figure 65: Evolution of distinct prefixes announced at TunIXP (TN).

At TunIXP, 94.44% of the prefixes are seen in 2014 and a 100% in 2015. Hence, as SIxP, it is growing since 2014.

Consequently, the newest IXPs (from 2013 onwards) are TunIXP, SIxP, NIXP, MIXP and DINX. However, taking into account their date of launch, the newest IXPs are TunIXP, SIxP, and DINX. Therefore, they are still growing. Although JINX and KIXP are the IXPs who have been collecting peering data the earliest, it seems that the peering with KIXP route-collectors will be more active in the end of the year, since it will not be logical a drastic drop like the one shown in its graph in 2015.

6.2 ASNs growth statistics per year at IXPs in PCH dataset

The goal of this experiment is to know whether AS allocation by AFRINIC and national registries are used and visible per year in PCH dataset.

The experiment is going to be divided into three parts. The first part will be the algorithm's explanation that returns the visible ASNs collected in PCH boxes per year, the second one will show the extra resources needed to accomplish the graph and the third one will contemplate the results.

6.2.1 Algorithm

Given the script used in item 5.4, and taking into account that we created a dictionary per IXP with the years as keys and the ASes announced per year as values, we can easily create a file with this dictionary called *year_ASN* without generating another script. Once all the ASNs are extracted for an IXP, we ensure that we have not repeated numbers for each year converting the list into a set. Then we write a file per IXP with the distinct years, followed by a comma and the length of the set per line into the document.

Last but not least, we could also have implemented this experiment with another script if the ASNs and the years were stored as a unique tuple of values in the database. In this case, we would need two tables, one for the origin ASes and another one for all of them. We just reused the code because we want to save as many space in the server as possible.

6.2.2 Extra resources

Since we are going to compare the number of distinct ASNs (both origin and all ASes) in this subpart, we do not include any information of the route-collectors in this item. Therefore, we are going to directly analyze the growth at the IXPs we know. Considering this, we can reuse the scripts created for MATLAB in the previous item (section 6.1.2), but changing the axes labels.

6.2.3 Results and graphs

We focus our study on the origin ASes per IXP with the following images.

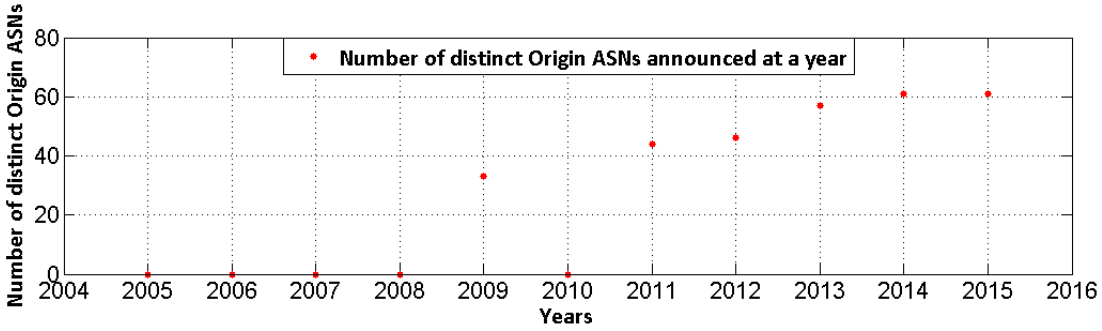


Figure 66: Evolution of distinct Origin ASes announced at CAIX (EG).

At CAIX, the first year that ISPs are peering with PCH is 2009. Actually, 47.14% of the visible origin ASes at the IXP are announced in that year. There is a drop to 0% of visible ASes in the next year, and then the percentage increases from 62.86% in 2011 to 87.14% in the last two years.

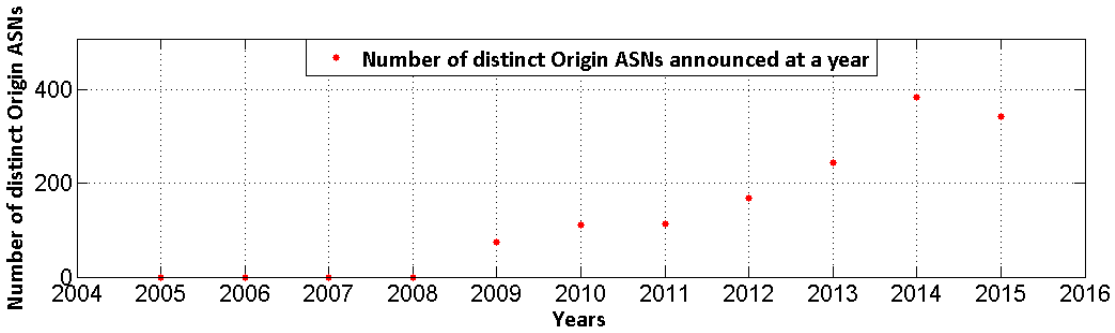


Figure 67: Evolution of distinct Origin ASes announced at CINX (ZA).

At CINX, the first year that ISPs are peering with PCH is also 2009 and only 15.56% of the origin ASNs visible at the IXP are announced in that year. There is no drop of visible ASNs till 2015, from where we have not the complete year information, and then the percentage increases to 79.67% in 2014.

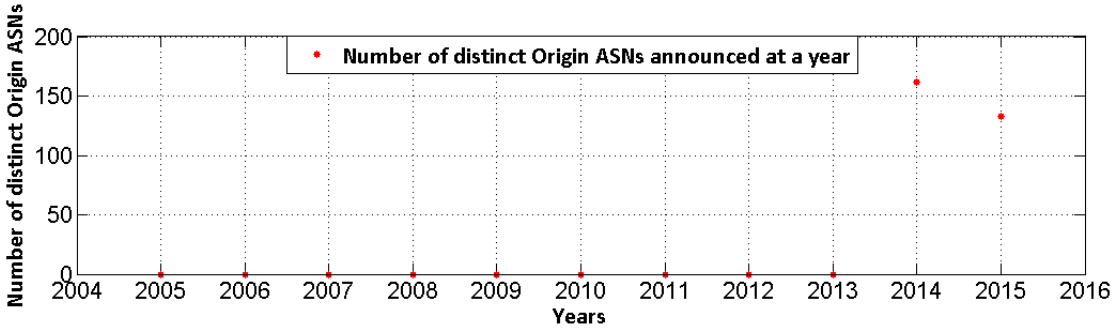


Figure 68: Evolution of distinct Origin ASes announced at DINX (ZA).

At DINX, there are only two years of peering data at PCH. The first year of ASNs peering data is 2014 and there is a coincidence of 92.57% origin ASNs visible at the IXP that are announced in that year. From 2015, we have a percentage of coincidence of 76% ASNs.

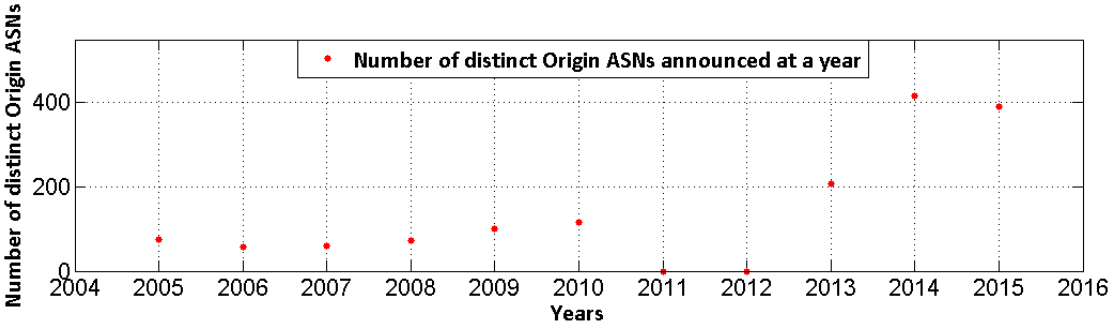


Figure 69: Evolution of distinct Origin ASes announced at JINX (ZA).

At JINX, 2005 is the first year that we found ASes peering data, and only 14.37% of the visible origin ASes are announced in that year. There is a two years drop to 0 visible ASNs in 2011 and 2012. Next, the percentage increases to its maximum that is an 80.97% in 2014.

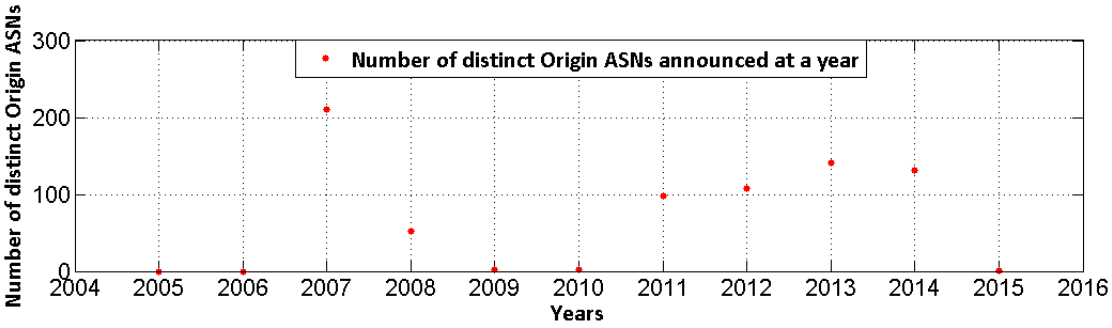


Figure 70: Evolution of distinct Origin ASes announced at KIXP (KE).

At KIXP, the first year that ASes are collected in PCH is 2007 and 58.29% of the origin ASNs visible at the IXP are announced in that year, which is the maximum percentage of origin ASNs announced at KIXP. There is an important two years drop to 2 visible ASNs in 2009 and 2010. Afterwards, the percentage is approximately maintained at 37.57% in 2013 and 2014. As we said in the previous item, the activity on the last year should be not registered yet.

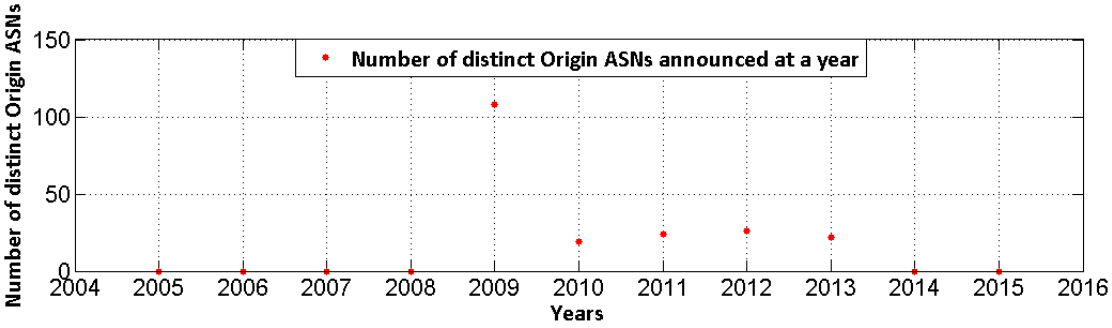


Figure 71: Evolution of distinct Origin ASes announced at MIX (MZ).

At MIX, the maximum percentage of origin ASNs announced is 83.72% in 2009. Then, it is decreased to a ratio of 17.64% in the following four years, till 2014 where PCH did not collect any data.

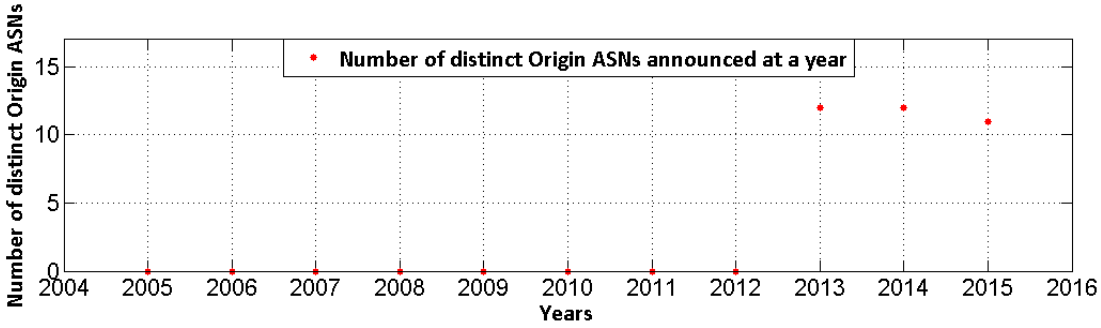


Figure 72: Evolution of distinct Origin ASes announced at MIXP (MW).

At MIXP, 92.31% of origin ASNs are announced in 2013, the first year that the IXP has been peering according to PCH and the percentage remains the same in the next year. In 2015, the percentage has decreased to 84.62% of origin ASNs.

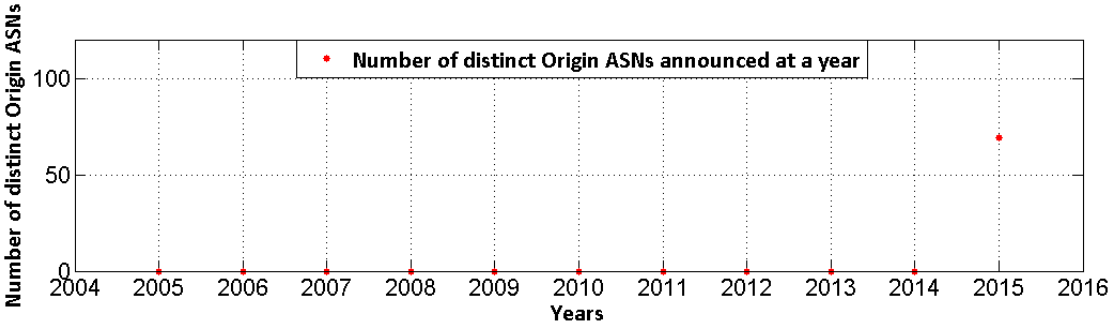


Figure 73: Evolution of distinct Origin ASes announced at NIXP (NG).

NIXP has only peering data this year, so that there is a coincidence of 100% about the origin ASNs announcement.

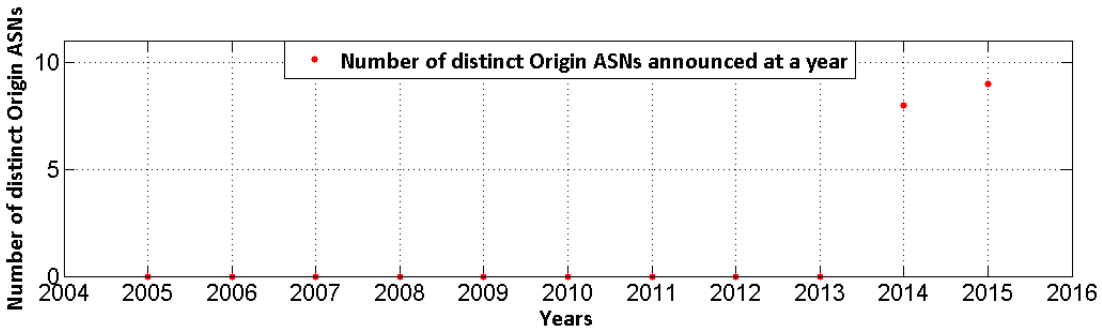


Figure 74: Evolution of distinct Origin ASes announced at SIxP (SD).

At SIxP, 88.89% of origin ASNs are announced in 2014, the first year that the IXP has been peering according to PCH. The next year, the percentage increases to 100%. It is clear that this IXP is growing nowadays.

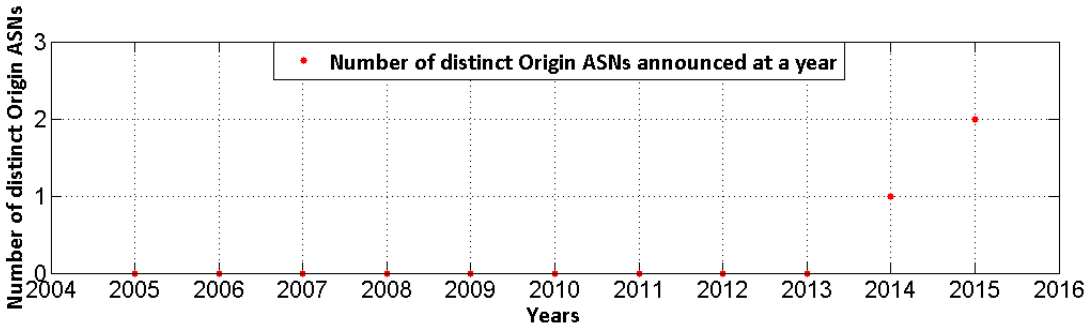


Figure 75: Evolution of distinct Origin ASes announced at TunIXP (TN).

At SIxP, 50% of origin ASNs are announced in 2014, the first year that the IXP has been peering according to PCH dataset. Afterwards, the percentage increases to 100% and because of that, it is clear that this IXP is growing nowadays.

In summary, according to PCH the newest IXPs (from 2013 onwards) are TunIXP, SIxP, NIXP, MIXP and DINX. However, taking into account their date of launch, the newest IXPs are TunIXP, SIxP, and DINX. Hence, they are still growing. Although JINX and KIXP are the IXPs who have been peering the earliest, it seems that the peering of KIXP will be more active in the end of the year, since it will not be logical a drastic drop like the one shown in its graph in 2015.

Finally, when we take a look at all the ASNs per IXP, we get similar results. There is just a slight variation with respect to the percentages of the ones given for origin ASNs. Since it is not adding any relevant information and in the graphs the difference is not appreciated, we will not add these graphs to maintain a fair extension in the thesis.

6.3 Unique number of Origin ASNs that appear at an IXP in consecutive and non-consecutive years from 2005 to 2015 in PCH dataset

The goal of this experiment is to know whether AS allocation by AFRINIC and national registries are used and visible in consecutive and non-consecutive years according to PCH dataset. Note that the ASes that appear only in one year are considered as ASes not seen every year, regardless if the announced year is 2015.

The experiment is going to be divided into two parts. The first part will be the algorithm's explanation that returns the visible and non-visible ASNs collected in PCH boxes every year and the second will contain the results.

6.3.1 Algorithm

Reusing the code written for extracting the origin ASNs and, thanks to the dictionary that stored the ASNs as keys and the list with the years where they are announced as values, we can extract two files:

- Unique list of ASNs that appear consecutively over time.
- Unique list of ASNs that do not appear consecutively over time.

Note that we consider that an ASN with only one year in its list, it is not seen every year, regardless of the year. Then, our first check after creating the dictionaries is to compute the number of appearing years for each ASN. If it is lower than one, then we will start creating two files:

- Unique list of ASNs that do not appear consecutively over time. We will keep track of the ASes inserted with an independent list.
- List of ASNs not seen in consecutive years and the year where they appear. This file can contain repeated ASNs since it is done for checking easily if the output of the unique list is the desired one.

By the same token, two files will be created for the ASes seen every year.

However, if the number of appearing years for each ASN is bigger than one, we check either if the consecutive year is in the list or if the actual year checked is 2015, and the previous one was 2014. If this condition is not satisfied, then it is not an ASN seen in consecutive years and we have also to store it in the non-consecutive list. With all these conditionals, we make sure that the ASes are added to the correct file. For each one of these cases, we also check before appending the ASNs to the file that will contain the unique list of ASes, if they are already in the independent list.

We could have appended the unique list of ASes with a set instead of a list, but we were interested in keeping the order of AS insertion for verifying if there was any mistake in the code. We could have done this code considering that if the last year was 2015 and we only had that value for an ASN, as a consecutive year. Besides, the two extra files (list of ASNs seen in non-consecutive and consecutive years, and the year where they appear) are not needed for our research, but it is

good to keep that information for future considerations or guesses. Also, we might have done this without reusing the code, but utilizing the files created for items 5.4 and 6.2, where the ASNs and the years were they are announced are stored.

6.3.2 Results

Thanks to the previous study of the distinct origin ASNs visible at the IXPs, we were able to check that the sum of ASNs seen in consecutive and non-consecutive years per IXP coincides with the number of visible ASNs computed in item 5.4. (see figure 20).

CC	IXP	Number and ratio of visible origin ASNs announced in consecutive years
ZA	JINX	436 (80.60%)
ZA	CINX	405 (81.00%)
KE	KIXP	149 (27.60%)
ZA	DINX	120 (67.80%)
EG	CAIX	34 (45.95%)
MZ	MIX	25 (18.94%)
MW	MIXP	12 (85.71%)
SD	SIxP	8 (88.89%)
TN	TunIXP	1 (50.00%)
NG	NIXP	0 (0.00%)

Table 22: Number and ratio of distinct visible origin ASNs announced in consecutive years per IXP.

CC	IXP	Number and ratio of visible origin ASNs announced in non-consecutive years
KE	KIXP	213 (72.40%)
MZ	MIX	104 (81.06%)
ZA	JINX	79 (19.40%)
ZA	CINX	77 (19.00%)
NG	NIXP	69 (100.00%)
ZA	DINX	55 (32.20%)
EG	CAIX	36 (54.05%)
MW	MIXP	1 (14.29%)
SD	SIxP	1 (11.11%)
TN	TunIXP	1 (50.00%)

Table 23: Number and ratio of distinct visible origin ASNs announced in non-consecutive years per IXP.

We can see that for the IXPs CAIX and TunIXP the ASNs announced are equally distributed between the consecutive and non-consecutive tables. However, the difference is noticeable for the consecutive percentages at SIxP (88.89%), MIXP (85.71%) and CINX (81%) with respect to the total number of origin ASNs visible at the IXP.

From the point of view of the ASNs not seen in non-consecutive years, on the one hand, we can remark NIXP (100%), since we only have peering data in 2015. On the other hand, IXPs like KIXP (72.40%) and CAIX (54.05%) give such results since they are growing in the last years.

All the ASNs results are pretty much equal. They are not included in the thesis because they do not change the conclusions extracted from the origin ASes.

6.4 Unique number of prefixes that appear at an IXP in consecutive and non-consecutive years from 2005 to 2015 in PCH dataset

The goal of this experiment is to know whether prefix allocation by AFRINIC and national registries are used and visible in consecutive and non-consecutive years according to PCH dataset. Note that the prefixes that appear only in one year are considered as prefixes announced in non-consecutive years, regardless if the announced year is 2015.

The experiment is going to be divided into two parts. The first part will be the key points changed in the previous algorithm's explanation in order to return the visible prefixes in consecutive and non-consecutive years collected in PCH boxes and the second will contain the results.

6.4.1 Algorithm

We reuse the code written in item 6.5 and the code generating the files in the previous item by replacing the ASN parameters by prefixes. The files and checks follow the same structure and the results are show in the next part.

6.4.2 Results

Thanks to the previous study of the distinct prefixes visible at the IXPs we were able to check that the sum of prefixes seen in consecutive and non-consecutive years per IXP coincides with the number of visible prefixes computed in item 5.3.

CC	IXP	Number and ratio of visible prefixes announced in consecutive years
ZA	JINX	5663 (58.44%)
ZA	CINX	3917 (72.38%)
EG	CAIX	3824 (56.51%)
KE	KIXP	2541 (60.00%)
ZA	DINX	1013 (55.78%)
MZ	MIX	638 (19.20%)
SD	SIxP	350 (70.56%)
MW	MIXP	54 (50.47%)
TN	TunIXP	17 (94.44%)
NG	NIXP	0 (0.00%)

Table 24: Number and ratio of distinct visible prefixes announced in consecutive years per IXP.

We can see that the top three IXPs that have more prefixes seen consecutively out of the total prefixes visible at the IXP are: CINX (72.38%), TunIXP (94.44%) and SIxP (70.56%). Moreover, we observe that the top three IXPs that have less prefixes seen in consecutive years out of the total prefixes visible at the IXP are: NIXP (since it is only peering in 2015), MIX (19.2%) and IBIXP (since we have not data of peering). JINX (58.44%), CAIX (56.51%) and MIXP (50.47%) are more or less equally distributed.

CC	IXP	Number and ratio of visible prefixes announced in non-consecutive years
ZA	JINX	4027 (41.56%)
EG	CAIX	2943 (43.49%)
MZ	MIX	2685 (80.80%)
EG	KIXP	1694 (40.00%)
ZA	CINX	1495 (27.62%)
NG	NIXP	1140 (100%)
ZA	DINX	803 (44.22%)
SD	SIxP	146 (29.44%)
MW	MIXP	53 (49.53%)
TN	TunIXP	1 (5.56%)

Table 25: Number and ratio of distinct visible prefixes announced in non-consecutive years per IXP.

6.5 Ratio of African ASNs assigned to the country that are visible at an IXP in PCH dataset

The objective of this experiment is to let regional (AFRINIC) and national registries know the ratio of African ASNs announced at PCH dataset that were assigned by AFRINIC.

This analysis is going to be divided into two different parts. The first part will contain the algorithm explanation that returns the percentage of ASNs assigned to the country and the second part will illustrate the results in two tables (one for the origin ASes and another one taking into account all of them).

6.5.1 Algorithm

Concerning this objective, we need to define as input data the continent, the CCs and the IXPs under study. Firstly, we create a folder where the results will be stored. Secondly, we give as a parameter the continent (AFRINIC), and we extract from the database the CCs and the IXPs of the continent inserting them in a dictionary structure as seen in figure 76.

We study the African origin ASes ratios and all the ASNs ratios at an IXP per CC. Therefore, we need to store the ASNs announced at each IXP in a dictionary as well as the ASNs allocated or assigned in AFRINIC region. On the one hand, the dictionaries *IXP_OriginASes* and *IXP_AllASes* have as keys the IXP's names and empty lists as values. On the other hand, we can start extracting the ASNs given a CC whereas we create the dictionary *CC_AFRINIC_ASes* with the CC as key and an empty list as value.

Regarding ASes in the RIRs database, they are stored in two different formats ("asplain" and "asdot"). [67] Then, we have to define a format to develop our analysis. Since most of the ASes are in "asplain" format, we apply this format to all of them. We have to take into account here that an ASN could be repeated in two formats and if so, we should not take them twice into account. We actually had stored twice some of them, since the result of the query in MySQL returned for example in 'ZA' 338 different ASNs, but if we put the same format in all of them, just 332 were different.

There are two ways to do convert from "asdot" format to "asplain":

```
asn = int(asn[:asn.find('.')])*65536 + int(asn[asn.find('.')+1:])
```

Or:

```
tab = asn.split('.')
asn = int(tab[0])*65536 + int(tab[1])
```

Both methods take the integer part of the number which will be multiplied by 65536. Then, the decimal part of the ASN is added as another integer to the result of the product done. Once the AS has the correct format, we insert it in the dictionary given the CC and we continue till we insert all the ASes extracted. We do this process for all the CCs stored.

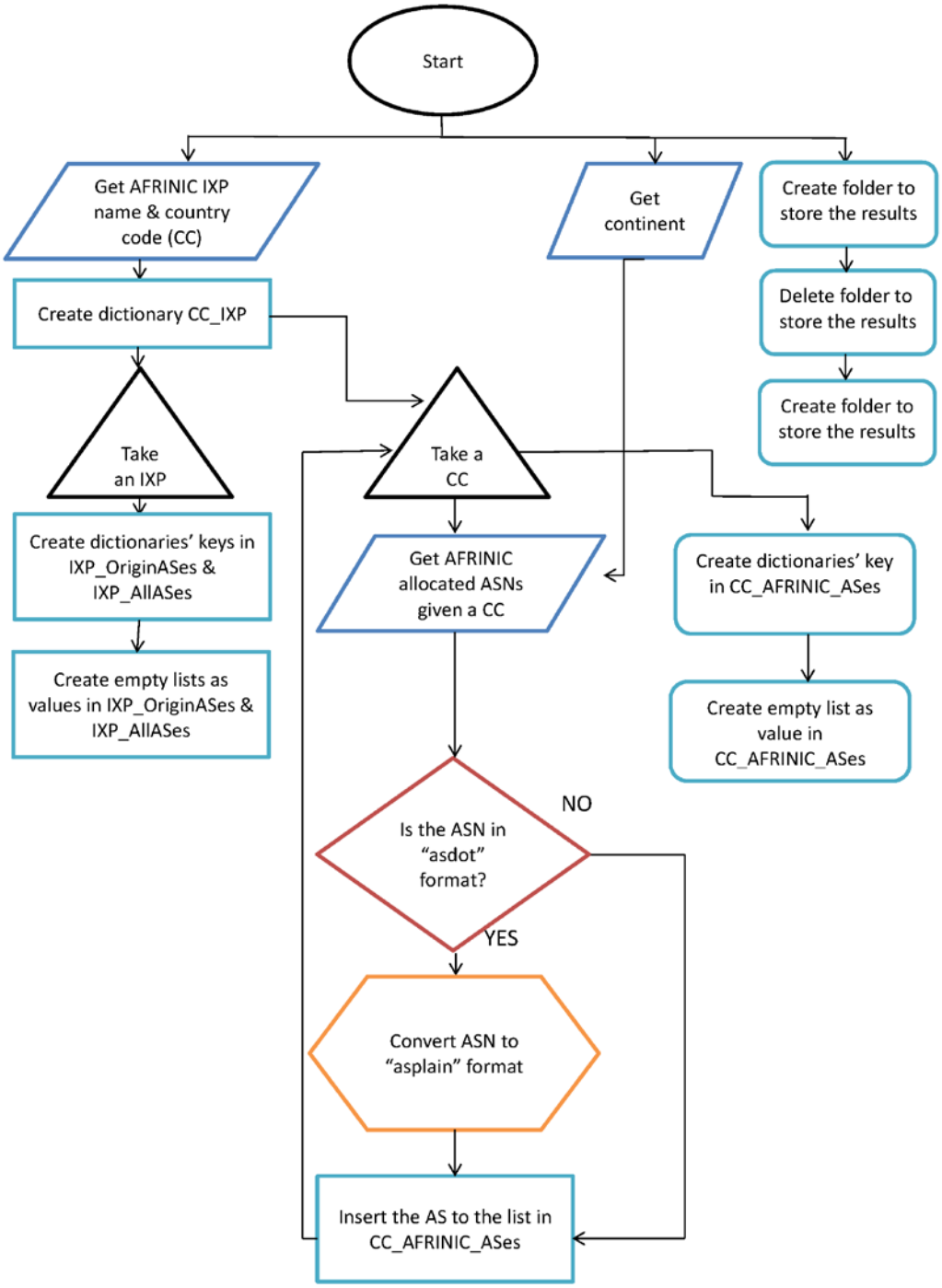


Figure 76: Descriptive flowchart with the input data formats and storage in memory.

After completing the list of ASNs in AFRINIC given the CC, we traverse the IXPs in the dictionary *CC_IXP*, in order to fill the dictionaries *IXP_OriginASes* and *IXP_AllASes*. Since we have calculated the distinct ASNs seen at an IXP over time in PCH data in item 4, we can generate these two dictionaries using the files where we store that information. Hence, we first check if there is a file in the path where these files are saved with the name of the IXP we are working with. However, there is an IXP that is no longer operational (IBIXP). Consequently, in the same conditional, we look

for a file that exists and whose name does not contain IBIXP. This could be also done by removing IBIXP from the dictionary where the IXP are called, but we want to show in the script later the ASNs seen at IBIXP in the AFRINIC region according to RIR database.

Then, if the file is found, we read every line till a comma is found, since the format of the document was: "ASN,route-collector,CC". For example, we extract the first lines in CAIX:

15475,route-collector.cai.pch.net,EG
6127,route-collector.cai.pch.net,EG
8452,route-collector.cai.pch.net,EG
21152,route-collector.cai.pch.net,EG

Once we take the AS value, we check in the list of ASes of *IXP_OriginASes* if it is already stored, so that we have only the non-repeated ones. We could have also done a set of these values, but working with lists, since they take into account the order of insertion, allowed us to easily correct an error by looking at the exact line we found it.

The dictionary *IXP_AllASes* is done right after completing the Origins one. We construct it in the same way, but looking at the file where all the ASNs seen at an IXP are stored. At this step, we have all the needed information for computing the intersection and the percentage of the ASNs assigned to the country that are visible at IXP.

We do this task by traversing again the *CC_IXP* dictionary per CC. Then for every IXP, we compute the difference and the intersection between the *IXP_OriginASes* of the IXP and the ASNs seen at the CC in AFRINIC. The difference is basically for checking the results obtained.

For computing the ratio, we divide the number of ASNs in the intersection over the number of ASNs seen at the country of the IXP in AFRINIC and we assure that this operation returns a float number. Finally we express that result as a percentage and we store it in a document. We could have also analyzed the results by showing them on the screen.

6.5.2 Results

Since the absolute values are low, the ratios are rounded in tables 26 and 27.

CC	IXP	African ASNs in the IXP's Country	Origin ASNs in IXP	Intersection	Difference	Ratio of African origin ASNs assigned to the country visible at the IXP (%)
SD	SIxP	7	9	7	2	100
KE	KIXP	80	362	59	303	74
EG	CAIX	69	70	49	21	71
MW	MIXP	12	13	8	5	67
ZA	JINX	332	515	199	316	60
MZ	MIX	26	129	15	114	58
ZA	CINX	332	482	182	300	55
NG	NIXP	144	69	52	17	36
ZA	DINX	332	175	43	132	13
TN	TunIXP	13	2	0	2	0

Table 26: Ratio of African origin ASNs assigned to the country visible per IXP.

On the matter of the top three ratios of IXPs whose visible origin ASNs are also assigned to the country of that IXP, we remark SIxP, KIXP and CAIX.

CC	IXP	African ASNs in the IXP's Country	ASNs in IXP	Intersection	Difference	Ratio of African ASNs assigned to the country visible at the IXP (%)
SD	SIxP	7	9	7	2	100
MW	MIXP	12	14	9	5	75
KE	KIXP	80	540	59	481	74
EG	CAIX	69	74	50	24	72
ZA	JINX	332	541	199	342	60
MZ	MIX	26	132	15	117	58
ZA	CINX	332	500	183	317	55
NG	NIXP	144	69	52	17	36
ZA	DINX	332	177	43	134	13
TN	TunIXP	13	2	0	2	0

Table 27: Ratio of African ASNs assigned to the country visible per IXP.

On the subject of the top three ratios of IXPs whose visible ASNs are also assigned to the country of that IXP, we remark that SIxP, KIXP and CAIX has almost the same ratio than for the origin ASes. However, MIXP ratio increases in 8% and therefore is the second IXP with more visible African ASNs. Finally, table 27 shows that some IXPs attract ISPs from other countries, creating hubs (e.g. JINX in ZA). In contrast, some IXPs such as NIXP, DINX and TunIXP, have a more national scope with varying degree of coverage (measured in percentage of national ASes connected to the IXP).

6.6 Number and ratio of ASNs by country assignment (local vs. external)

The goal of this experiment is to let Regional (AFRINIC) and National Registries know the ratio of ASNs announced worldwide at PCH dataset that were assigned by AFRINIC.

This analysis is going to be divided into three different parts. The first part will contain the algorithm explanation that returns the number of ASNs assigned to each country and region. The second one will show the extra resources needed to accomplish the graph. The third one will illustrate the results for the origin and all the ASNs collected, and we will explain the differences between them.

6.6.1 Algorithm

For our purpose, we can reuse almost completely the code written for the item 9 in this chapter. We should first change the way we create the dictionary *CC_AFRINIC_ASes*. In this case we are interested in knowing the distribution of ASes worldwide, so that we are going to create this dictionary with all the CCs that we extract from the database, not only the IXP's ones. Similarly, we create the remaining regions dictionaries: *CC_ARIN_ASes*, *CC_RIPE_ASes*, *CC_APNIC_ASes* and *CC_LACNIC_ASes*.

We are interested in the ratios worldwide, and since PCH does not peer with every IXP, we incorporated more input data from other databases. We decided to use PeeringDB, a database with information of networks interested in peering [68]. PeeringDB is freely available and it contains records about IXPs, ASNs and some others. The relation of tables for extracting the required parameters is shown in the next graph.

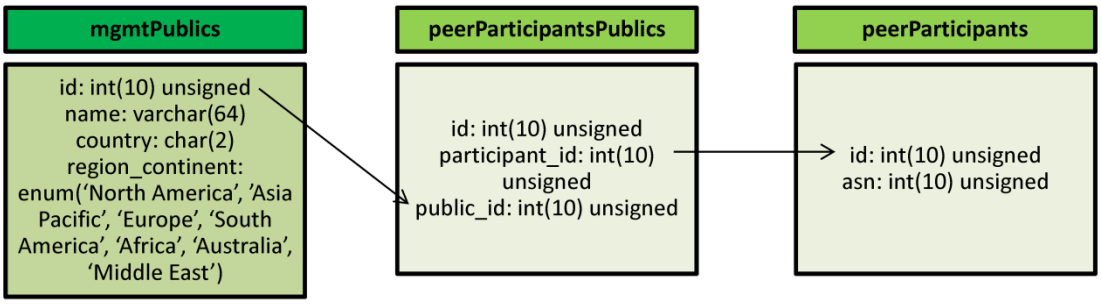


Figure 77: Relation of PeeringDB parameters between tables.

This database has extra information in the case of AFRINIC region because it contains some extra IXPs that we are not considering (AMS-IX-East-Africa, LUSAKA-IXP and ANGOLA-IXP) and therefore we have not enough information to compare them to. So, they will not be included in the study. Concerning the IXPs of interest, we realize that some were stored with a different name than the names in our database.

Thus, we had to establish that NIXP was the same as IXPN, MIZ was the same as MOZIX and MIXP was the same as MIX-BT in Peering DB before extracting the participant ids at the second table. Once we return the ASNs seen at an IXP, we insert them in *IXP_OriginASes* and in *IXP_AllASes*.

Now that we have all the input data defined, we are going to create the following dictionary structure:

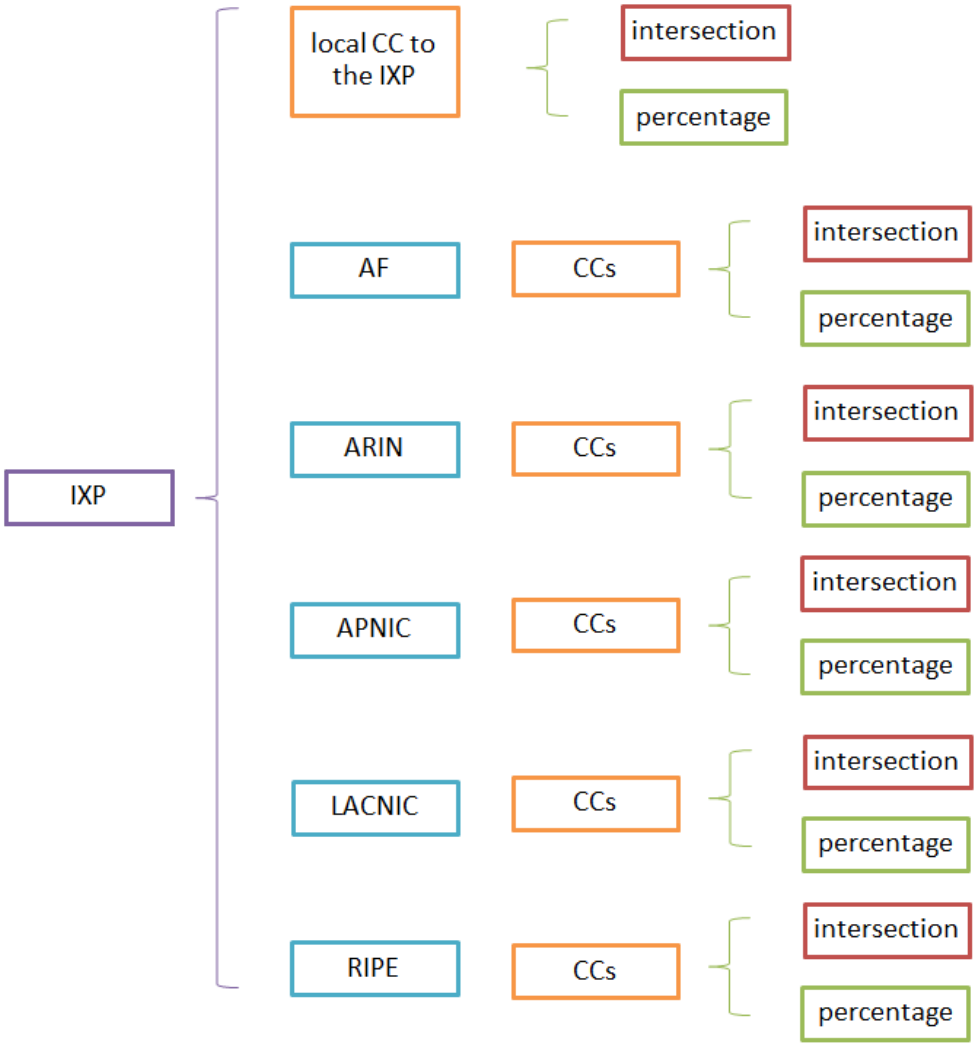


Figure 78: *Full_dict* structure.

It implies that we are going to have a dictionary (called *Full_dict*) with IXPs as a key. Then each IXP has as value a list of dictionaries: one is the local CC and the rest of them are the regions names (the remaining CCs will be organized per region). Next, every region will have another dictionary inside, with CCs as keys and as values a list with two dictionaries:

- A dictionary whose key is the word *intersection*. We will store the ASes of an IXP that are also allocated or assigned in the same country as the IXP according to RIRs. The format of intersection is a set, since we could compute the intersection straight-away in Python and we will also be sure that there are not repeated ASes.
- A dictionary whose key is called *percentage*. We will keep the value of that intersection over the total number of origin ASes collected at that IXP. We will do the same for considering all the ASes, but we are going to focus the explanation in the origin ones.

Note that the local CC will have also these two dictionaries.

Once the structure is defined, we will explain how it is built and filled. As it is shown in figure 79 (next page), every time we take an IXP, we create a local variable called *diff*, where we will store the non-classified ASes that we have at an IXP (i.e. the difference between the origin ASes of an IXP and the ASes allocated in AFRINIC at the CC of the IXP). It is possible that the intersection of ASes is empty, so that we define the percentage as 'NONE'.

Until now, we have just computed all the local intersections of an IXP and its ratios. We have to do the same for the external CCs in AFRINIC (i.e. the non-local CCs). In order to do so, we create the dictionary as mentioned and we do the intersection with *diff*, as we should have classified some of the ASes. Every time we have a valid intersection, we remove the new classified ASes from *diff* and we store the intersection and the percentage. We repeat this in all the regions.

After filling these dictionaries of an IXP, it is possible that there are some ASes still not classified in *diff*. Thus, we check each AS in diff according to the next table [69] and we create three lists:

- A list which will contain the ASNs reserved: reserved.
- A list that will store the private ASes: private.
- A list with the remaining ASes not stored in reserved or in private: *NC*.

AS Number	Assignment
0	Reserved
1 – 48,127	Assigned
48,128 – 54,271	Not assigned
54,272 - 64511	Reserved for the IANA
64512 - 65534	Private range
65535	Reserved

Table 28: ASN ranges.

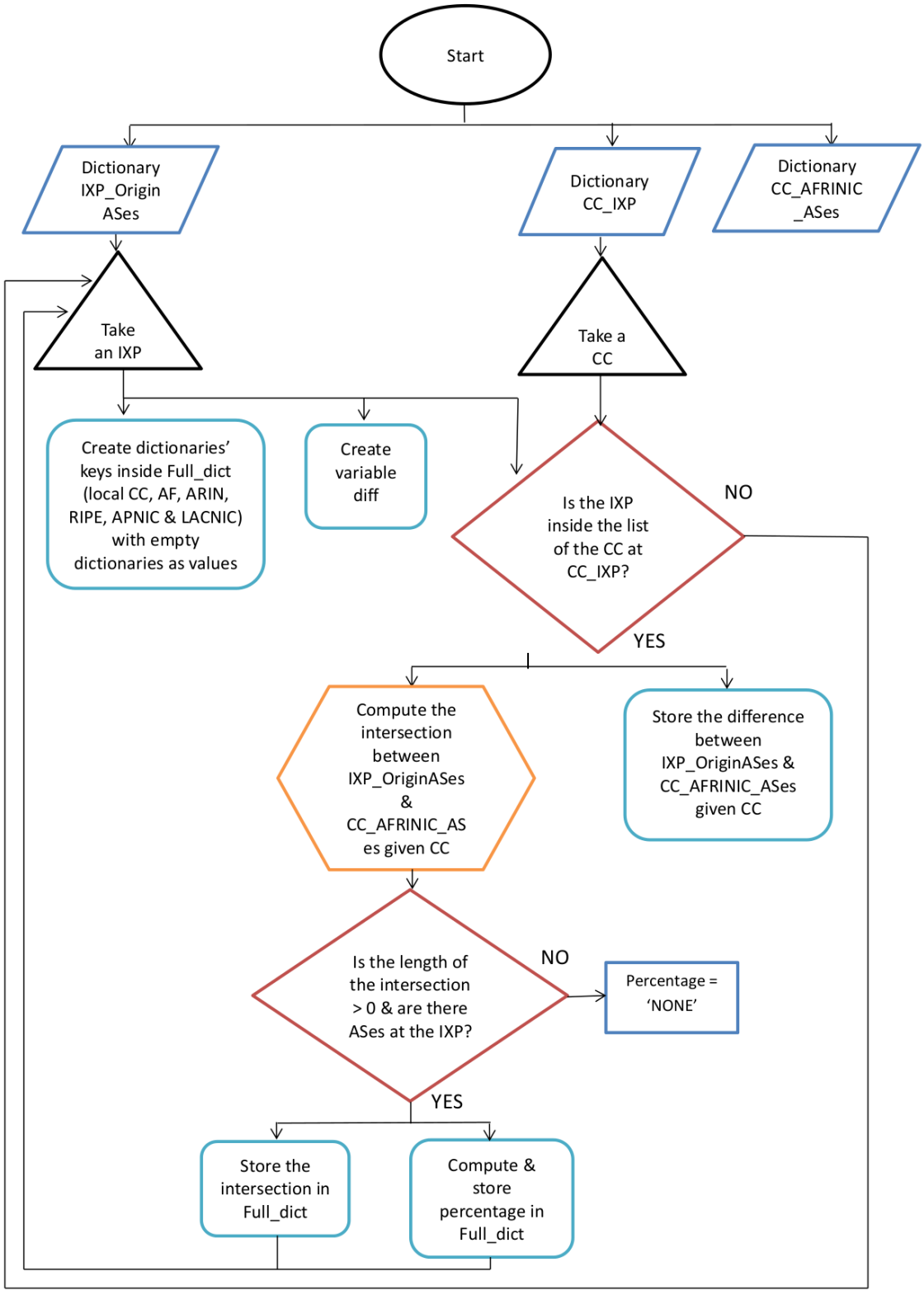


Figure 79: Descriptive flowchart with the input data formats and storage in memory for the local CC and the AFRINIC ones.

In any case, it might happen that there will be still ASNs non-classified, so that we define a method that extracts the CC and the region of an AS by means of the *whois* command. Then for each AS we store the new CC and the region in auxiliary variables (*aux_cc* and *aux_region* respectively). We create also an auxiliary variable with the ASN as a set (*AS_set*), in order to make unions and differences easily. Besides, we are going to compare the local CC with the new one (so that we insert it in the correct dictionary). In order to do so, when we detect the local CC, we store it in a variable called *current_cc*.

We first check if the CC in the auxiliary variable is the same as we had in our previous CC (also stored in an auxiliary variable called *current_cc*). If so, we add this new AS to the intersection of the local CC, we perform the same operations to compute again the percentage and we remove it from *NC* list.

If the CC is not the local one, we must check the region in *aux_region*. For the AFRINIC region case, we have to check if the CC is already inside the dictionary of that region. If so, we do the union between *AS_set* and the intersection at that CC, we calculate the new ratio and we delete the AS from the *NC* list. However, it could occur that the CC is not yet in this dictionary, in such a case, we create a dictionary with the new CC as key and as values the dictionaries where the intersection and the percentage will be stored. It was really important to detect that the CC returned by the method could be 'ZZ' because this code is used for the reserved ASes in AFRINIC region.

The rest of the regions are checked as follows:

- We get the region of the AS.
- We check if the CC is not in the list of CCs of that region or if the percentage assigned to that CC is 'NONE'. If so, we create a new dictionary for that CC and we fill it as before. If the AS is reserved, the CC is a special character that we define as a blank. If the percentage is 'NONE' but the CC is in the region country code list, it means that has been allocated after we collected our data.
- If the CC is in the list of CCs of the given region, we compute the union between the intersection stored and the new AS, the ratio as always and we remove the AS from *NC* list.

Next, the percentage of each CC, except the reserved ones, is added to have the ratio in a region. The local percentage is not added to the AFRINIC one, so that we can compare between the local and the external ratios in AFRINIC. Besides, if any AS remains unclassified, we will put it in the graph, as well as the reserved and private ASes.

We could have calculated the ratios by adding them to a local variable every time we calculate a new intersection. Nevertheless, since some ASes could have not been classified because they are reserved or they were not allocated when we collected the data, with this approach we should remove the previous percentage and add the new one to the *percentage* of the region. Therefore, it would be error prone and we avoided them with our implementation.

Finally, the results could be given in ratios or in number of ASes. We chose the last option, as we were able to directly plot the ratios with them.

6.6.2 Extra resources needed

Since PeeringDB is frequently updated, we programmed the table downloading and its insertion in the database every day at a fixed time thanks to a cron job, so that we could analyze if any AS is allocated to another region or reserved to any in our results.

Lastly, regarding the graphs, we decided to illustrate the results in 3D Pie Charts. In order to do so, we wrote some HTML code (HyperText Markup Language) similar to the example given at [70]. We just changed the title, the dimensions of the graph, and we put the number of ASes given the classification made with their corresponding labels per IXP.

6.6.3 Results and graphs

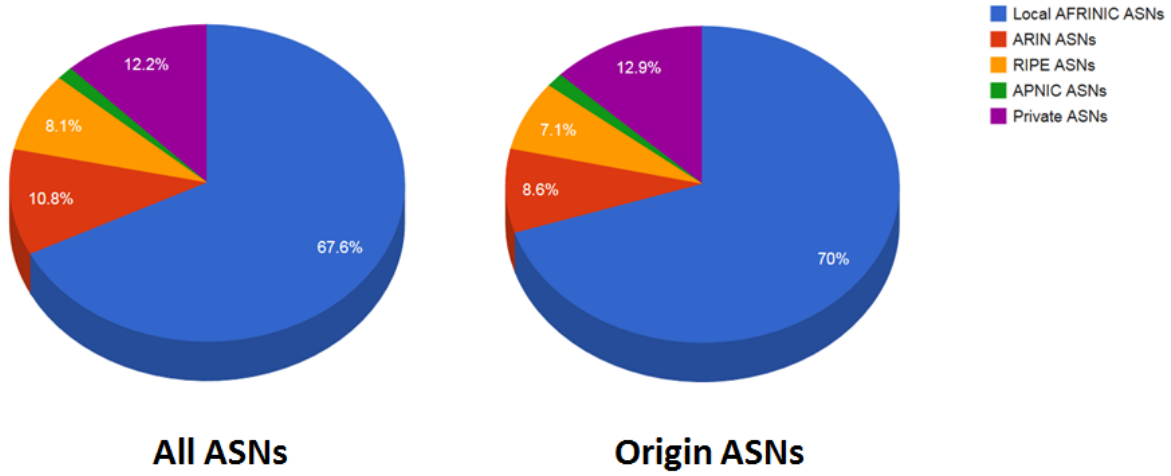


Figure 80: Ratio of ASNs by country assignment in routes collected by PCH boxes at CAIX (EG).

According to our dataset, 70% of the Origin ASNs announced in CAIX are local to the IXP under study. Then, among the rest of the regions, the region with most Origin ASNs is ARIN, followed by ARIN, RIPE and finally, by APNIC. However, 12.9% of the Origin ASNs are privates ones. We observe that the difference in the percentage of local and private ASNs to the IXP is almost the same when we consider all the ASNs seen at the IXP, not only the origin ones. However, when we consider all the ASNs, the percentage for RIPE and ARIN has a higher ratio than the ratio with the origin ones, and at APNIC region has not changed.

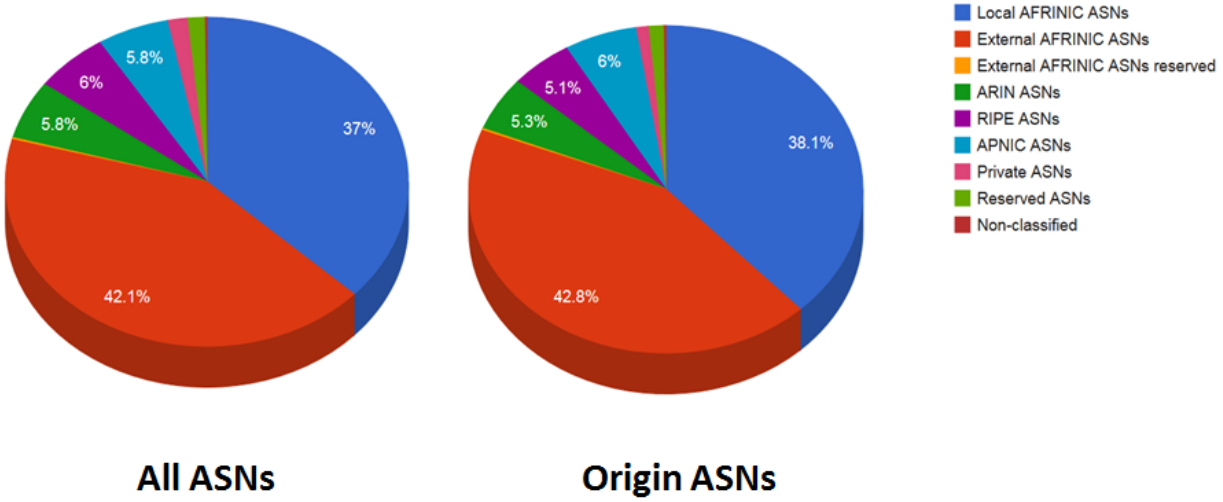


Figure 81: Ratio of ASNs by country assignment in routes collected by PCH boxes at CINX (ZA).

According to PCH dataset, 38.1% of the Origin ASNs announced in CINX are local to the IXP under study. It is important to notice that 42.8% of the ASNs are from external countries of the AFRINIC region and that a small percentage (0.2% - an AS) is reserved for the AFRINIC region. There are not Origin ASNs visible in LACNIC from this IXP, but they are almost equally distributed in ARIN, RIPE and APNIC regions. The rest of the Origin ASNs are privates (1%), reserved (1.6%) and non-classified (0.2%). As in CAIX, the percentages per region has varied a bit in Pie Chart left, but not for the private, reserved and non-classified ASNs. The total ASNs ratio corresponding to the AFRINIC region is slightly lower than the ratio considering the Origin ASNs, and the region with biggest ratio, without considering the AFRINIC region, has changed from APNIC to RIPE.

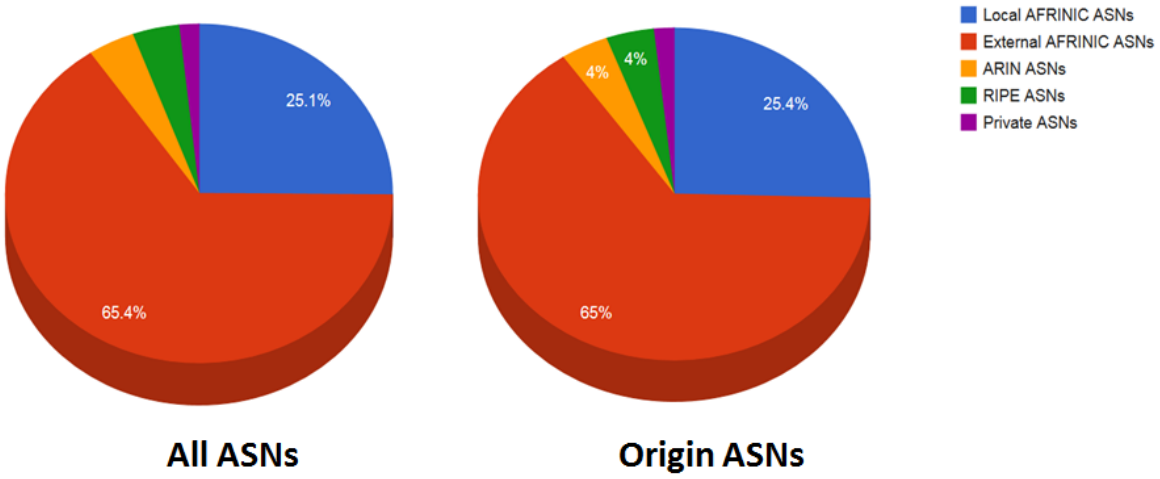


Figure 82: Ratio of ASNs by country assignment in routes collected by PCH boxes at DINX (ZA).

According to our dataset, 25.4% of the Origin ASNs announced in DINX are local to the IXP studied. In addition, 65% of the Origin ASNs are external to the IXP country, but they are in the AFRINIC region. However, among the rest of the regions, ARIN and RIPE are equally distributed, but there is a 0% percentage in LACNIC. Just 1.7% of the Origin ASNs are reserved. In Pie Chart left, the local ASN percentage is 0.3% lower and the external percentage in AFRINIC is 0.4% larger.

Therefore, the remaining changes in the percentages come from the ARIN and RIPE ASNs. The private ASes remain the same as expected.

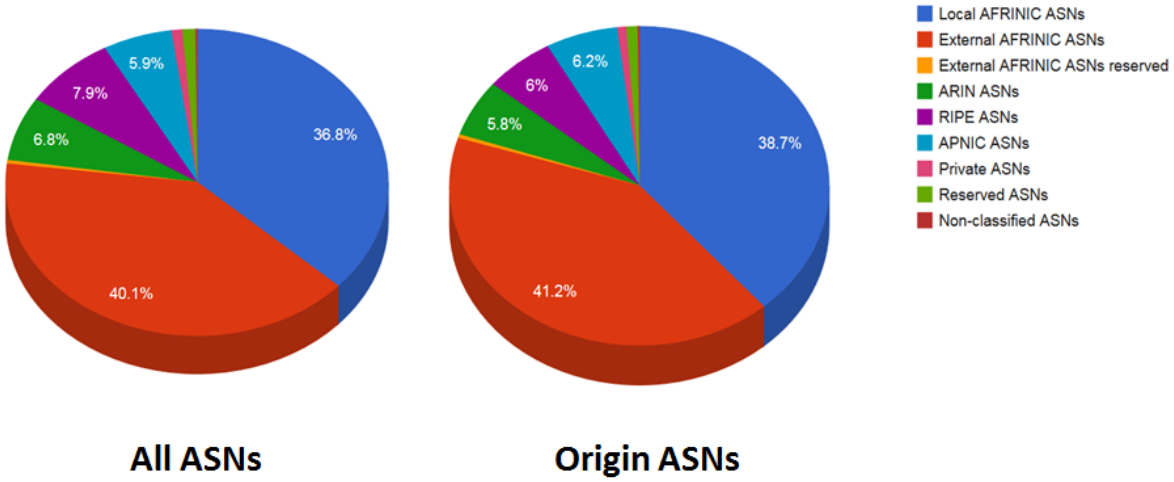


Figure 83: Ratio of ASNs by country assignment in routes collected by PCH boxes at JINX (ZA).

The graph right indicates that 38.7% of the Origin ASNs announced in JINX are local to the IXP, 41.2% of them are external to the country, but still in the AFRINIC region, and just a 0.4% of the ASNs is reserved to AFRINIC. There are not Origin ASNs visible at LACNIC at this IXP, but it is almost equally distributed in ARIN, RIPE and APNIC regions. We also observed 0.8% of private ASNs, 1 % of reserved ASNs and 0.2% of non-classified ASNs. Similarly to the previous case, the percentages have slightly been modified in Pie Chart left. The percentage corresponding to the AFRINIC region is 1% lower, except for the reserved one. The external region with most ASNs is RIPE now, followed by ARIN and APNIC. The rest of the percentages remain the same.

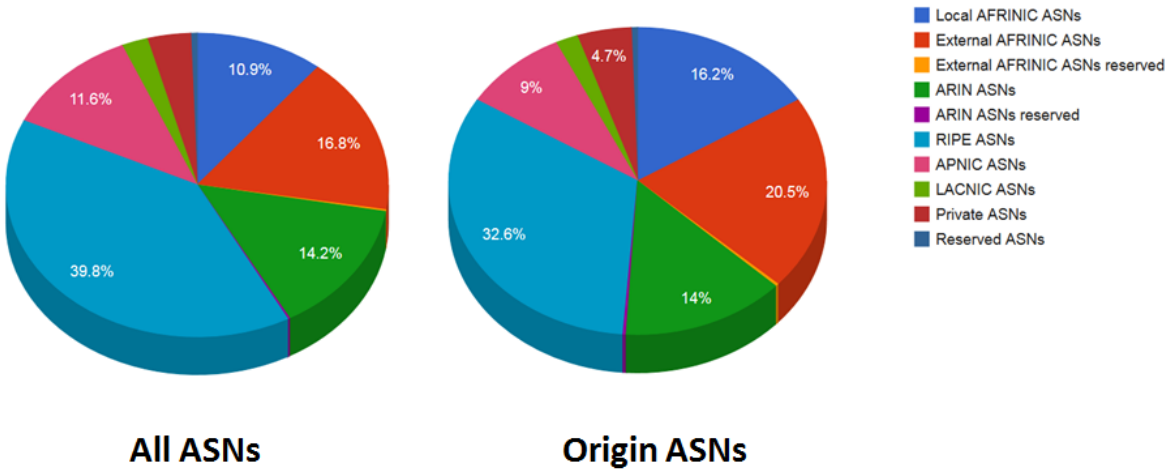


Figure 84: Ratio of ASNs by country assignment in routes collected by PCH boxes at KIXP (KE).

KIXP is the most specific IXP. We have 16.2% of Origin ASNs local to the IXP, 20.5% external to the country, and 0.3% of them are reserved external in AFRINIC. ARIN, in this case, also has a

percentage of 0.3% of Origin ASNs reserved and 14% that are already allocated or assigned to the region. Moreover, we found assigned 32.6% of them to RIPE, 9% to APNIC and 1.9% LACNIC. This is the only IXP that has both reserved ASNs for any other region that is not AFRINIC and assigned ASNs to LACNIC. Finally, we found that 4.7% of the Origin ASNs were private and 0.5% reserved. However, when we considered all the ASNs, the percentage of local ASNs to the IXP is 5.3% lower and the ratio of external ASNs in the AFRINIC region is 3.7% larger. RIPE is still the external region with the largest number of ASNs, followed by ARIN, APNIC and LACNIC, as before. The private ASNs ratio is reduced 1% and the LACNIC ASNs is 2.2%. Obviously, the reserved ASNs remain the same.

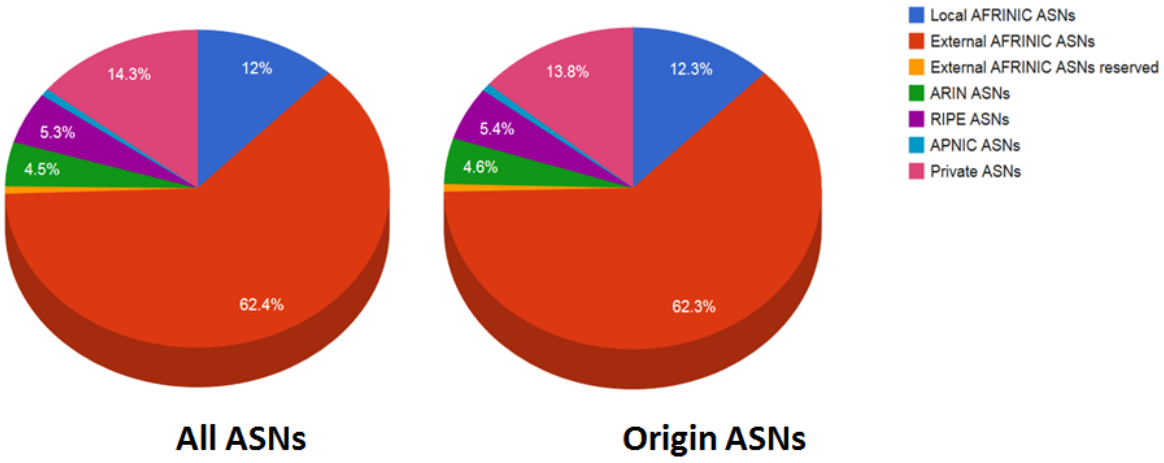


Figure 85: Ratio of ASNs by country assignment in routes collected by PCH boxes at MIX (MZ).

At MIX, we have a huge amount of external AFRINIC ASNs (62%) with respect to the 12% of ASNs that are local to the country of MIX. The Pie Chart right shows that 0.8% of Origin ASes are reserved to AFRINIC. Besides, we found 5.4% of Origin ASNs in RIPE, 4.6% in ARIN and 0.8% in APNIC. The rest of them are private ASNs. The biggest change when comparing all the ASes collected is located in private ASNs, since we are considering more ASNs, it is 0.5% larger. The rest of the percentages are mostly the same in both graphs.

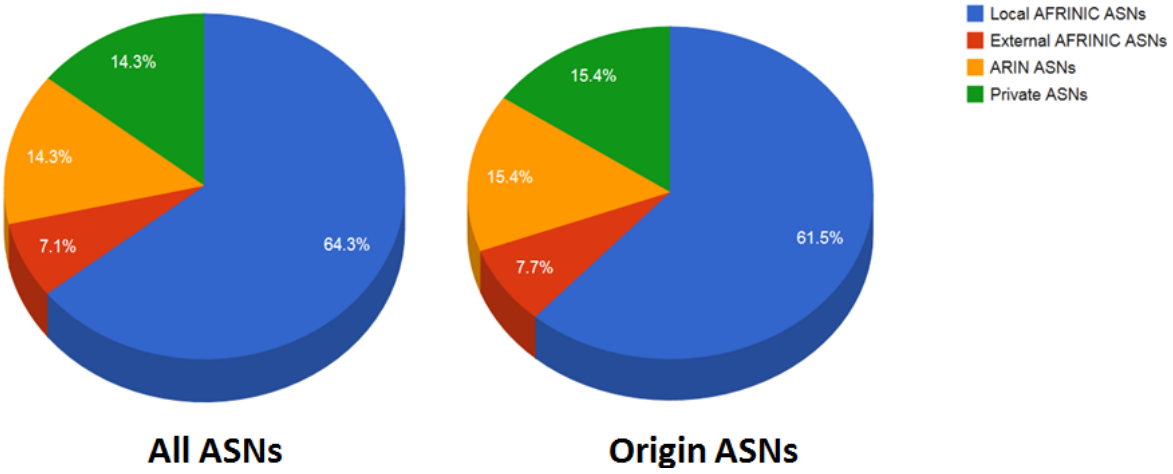


Figure 86: Ratio of ASNs by country assignment in routes collected by PCH boxes at MIXP (MW).

According to our dataset, 61.5% of the announced Origin ASNs in MIXP are local to the IXP. Then, among the rest of the regions ARIN is the region with more Origin ASNs assigned or allocated (15.4%). The same percentage is given at private ASNS and just 7.7% of the ASNs are external to the AFRINIC region. Besides, the percentage of local ASNs at MIXP is 2.8% larger than the ratio of Origin ones, but is lower in the percentage of external AFRINIC ASNs to the IXP. This is due to the new total number of ASNs under study in MIXP. The rest of the percentages are 1.1% lower than before.

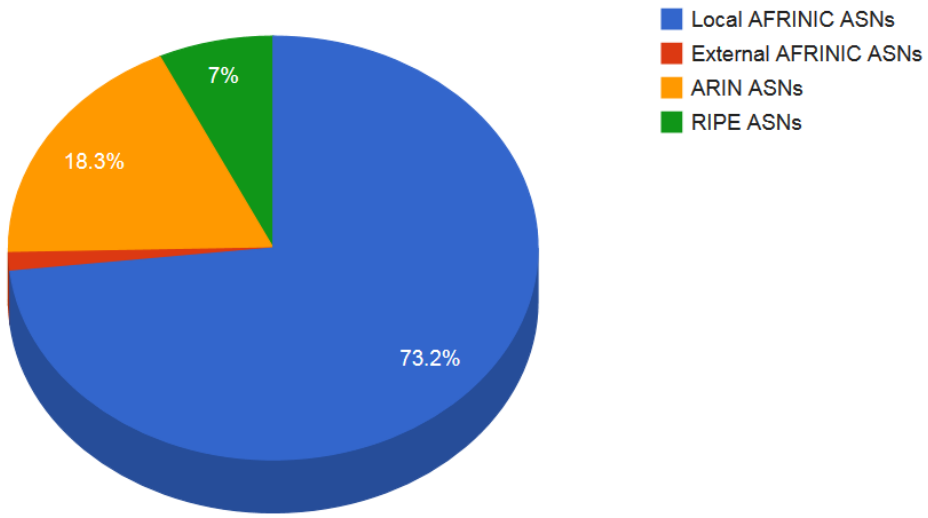


Figure 87: Ratio of ASNs by country assignment in routes collected by PCH boxes at NIXP (NG).

In NIXP, 73.2% of the ASNs are local to the IXP, and 1.4% of the ASes are external to country of NIXP (Nigeria). The next region with the largest number of ASNs is ARIN, followed by RIPE. All the ASNs were classified in this case. Note that in this case, all the ASNs under study are Origin ones, and therefore there is just one Pie Chart.

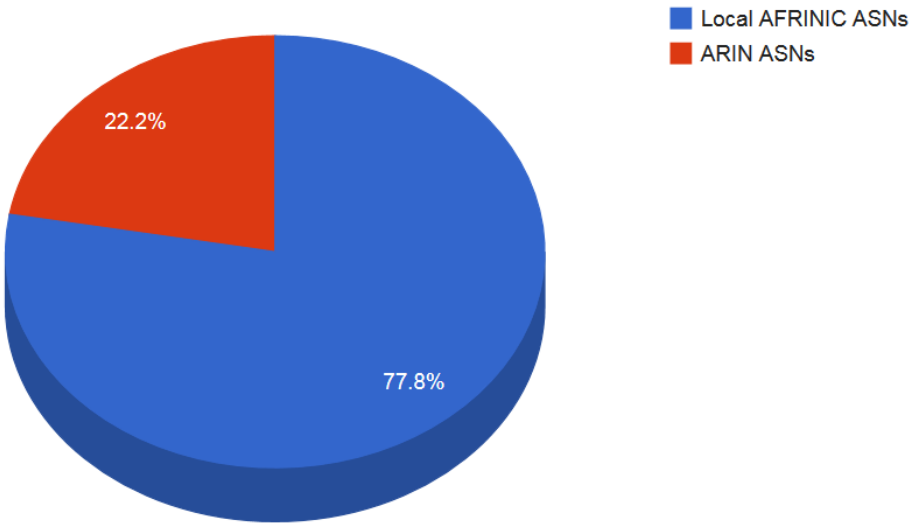


Figure 88: Ratio of ASNs by country assignment in routes collected by PCH boxes at SIxP (SD).

At SIXP, we find that 77.8% of the ASNs are local to the IXP, and the remaining ratio of ASNs is found in ARIN region. Similarly to the previous case, all the ASNs studied are Origin ones, and therefore there is just an illustrating graph.

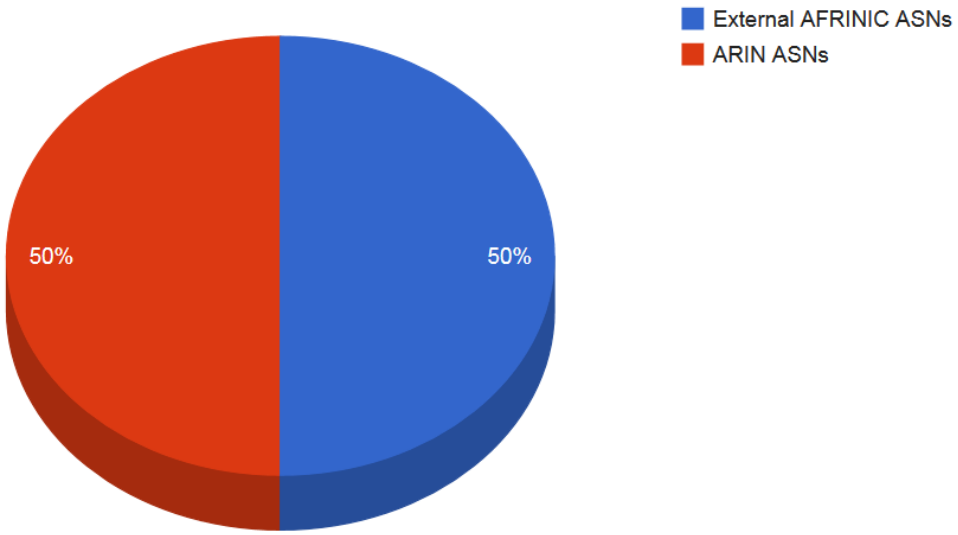


Figure 89: Ratio of ASNs by country assignment in routes collected by PCH boxes at TunIXP (TN).

At TunIXP, there was an ASN in AFRINIC but not local to Tunisia, and another one that came from ARIN region. As in the previous two cases, there is no difference between the Pie Charts considering all the ASNs and the Origin ASes.

In summary, the top four IXPs ratio of ASNs local to the country are SIxP (77.8%), NIXP (73.2%), CAIX (67.6%) and MIXP (64.3%). The four highest IXPs percentage of external AFRINIC ASNs are DINX (65.4%), MIX (62.4%), TunIXP (50%) and CINX (42.1%).

Next, we focus on the top three IXPs ratio of ASNs that are not in the AFRINIC region since the fourth IXP does not present a significant ratio. The three highest IXPs percentage of ARIN ASNs are TunIXP (50%), SIxP (22.2%) and NIXP (18.3%) and MIXP (14.3%). Regarding the top three IXPs ratio of RIPE ASNs we find KIXP (39.8%), CAIX (8.1%) and JINX (7.9%). In addition, the top three IXPs ratio of APNIC ASNs are KIXP (11.6%), JINX (5.9%) and CINX (5.8%). Last but not least, we find that only one IXP has LACNIC ASNs which is KIXP (14.2%).

Chapter 7: Conclusions

The main goal of this project was to provide diverse statistics based on historical routing data collected from African IXPs that would be helpful for some Institutions to make suitable decisions and empowering the Internet in the AFRINIC region by a better understanding of the underlying relationships. To achieve this goal, we collected the PCH routing information about how local ASes have been peering over time in the different regions.

Once the dataset was defined, we used Python to download and classify the dataset. While downloading PCH data, we geolocated all the route-collectors. Then, we parsed all the data in parallel and stored it on a server for further processing.

Since we wanted to analyze how African prefixes and Autonomous Systems (ASes) are seen from other countries, we also needed the information contained in the RIRs (Regional Internet Registry) databases. So we added the RIR database provided by the research team.

After all the required data was stored in the server, we performed the computations required for the distinct statistics in Python. We represented the results in MATLAB graphs and Pie Charts via HTML code. As some results did not need a graph, we used tables instead. It can be asserted that the initial requirements have been fulfilled since our results show that 95.58% of the prefixes appear since their allocation date and 87.12% of them have appeared on 2015 as the year last of appearance. Moreover, the most frequent prefixes come from South Africa, Nigeria and Egypt. Also, the IXPs mostly used for peering (JINX and CINX according to our dataset) have more reallocated prefixes than the rest of the IXPs. In addition, the newest IXPs (from 2013 onwards) are TunIXP, SIxP, NIXP, MIXP and DINX. However, taking into account their date of launch, the newest IXPs are TunIXP, SIxP, and DINX. Hence, they are still growing, whereas the IXPs who have been peering the earliest (JINX and KIXP) show a drop in evolution.

While the thesis focuses on results, during the entire process we have followed different research lines and we have completed the ones that seemed the most promising. The most important difficulty found was to deal with so many different data sources, involving adaptations in format, interpretation and cleaning. Besides, we integrated many systems and languages to achieve the analysis, so our scripts were compatible among the different resources used.

However, not every IXP in Africa is covered by PCH dataset. Besides, PCH route-collectors are not deployed as soon as IXPs are launched. Although PCH has an open peering policy, not all ISPs at an IXP peer with PCH boxes. In addition, some route-collectors did not collect data every year up to 2015. Furthermore, it is important to take into account that for some prefixes allocated by other RIRs, the allocation dates in the AFRINIC delegated files are biased (e.g. 00000000, years 1984, 1989, 1990, etc.). In conclusion, this dataset is biased and this bias could be reduced incorporating additional routing information such as the RouteViews dataset.

Even though that the computed statistics allow some Institutions (e.g. ISOC or African Union) to make suitable decisions for establishing regional IXP, there are additional possible studies that will complete the contribution to the Peering growth on the African Internet. They have not been carried out since they are not under the project scope, but among them we emphasize:

- Number and ratio of prefixes by country assignment (local vs. external).
- Visible Bogon[3] announcements.
- Number and ratio of long prefix length announcements. This (i.e. Comparison of Member behaviors on aggregation and deaggregation compared between the Peering Point announcements and Transit/Internet upstream announcements.
- Ratio of IPv4 blocks assigned to the country that are visible at IXP
- Ratio of IPv6 blocks assigned to the country that are visible at IXP
- Comparison on the ratio of IPv6 and IPv4 prefixes.
- Ratio of 2 Byte vs 4 Byte ASNs.
- Ratio growth rate of ASNs and prefixes at an IXP over time.
- Distribution of non-local ASNs by country.
- ASN and prefixes Stability at an IXP.

Added to our results, they represent the IXP view. We also plan to compute different statistics based on data collected from *(i)* the set of IXPs in the country – National view, *(ii)* the set of IXPs in the region - Regional view *(iii),* the set of IXPs in the continent involved the PCH dataset – Global view.

Moreover, since PCH website has changed, the PCH data downloader will be reconfigured and it will be executed as a cron job in order to get the updated dataset. Consequently, the PCH data parsing and storage will be also reconfigured as a cron job, so that any IXP at which PCH peers will be automatically involved in the dataset. Our scripts are therefore going to be improved.

Finally, they are going to be integrated to the computational module of the web platform for "Inter-Domain IP routing economic analysis" developed by the research team in which the project was completed. Since the website will be publicly available in the near future, all the scripts must be executed periodically as cron jobs and update the statistics. Consequently, this web platform will be really helpful for taking suitable decisions aiming at empowering the Internet at any region. For instance, it would easily help ISPs to choose at which IXP to peer next, or be used by the stakeholders to evaluate the growth of their IXP in comparison with others. Moreover, the Internet Society would be able to determine as regional IXPs those at which we discovered most prefixes and origin ASes connected to and boost them.

[3] Bogon prefixes are private and reserved addresses.

Annexes

ANNEX A: Schedule

This final degree project followed a predefined order of milestone realization over the time it was developed. In this annex section we detail the milestones, their relation and precedence, as well as the Gantt chart of this project.

A.1. Milestones sequencing

This project has been developed in 7 months. The total number of hours required were 440 broken down in 7 milestones. The duration of each milestone is expressed in weeks instead of hours, since the Gantt chart is also expressed in weeks. Among others, the prerequisites of this project in terms of programming languages were Python, MATLAB, MySQL and HTML.

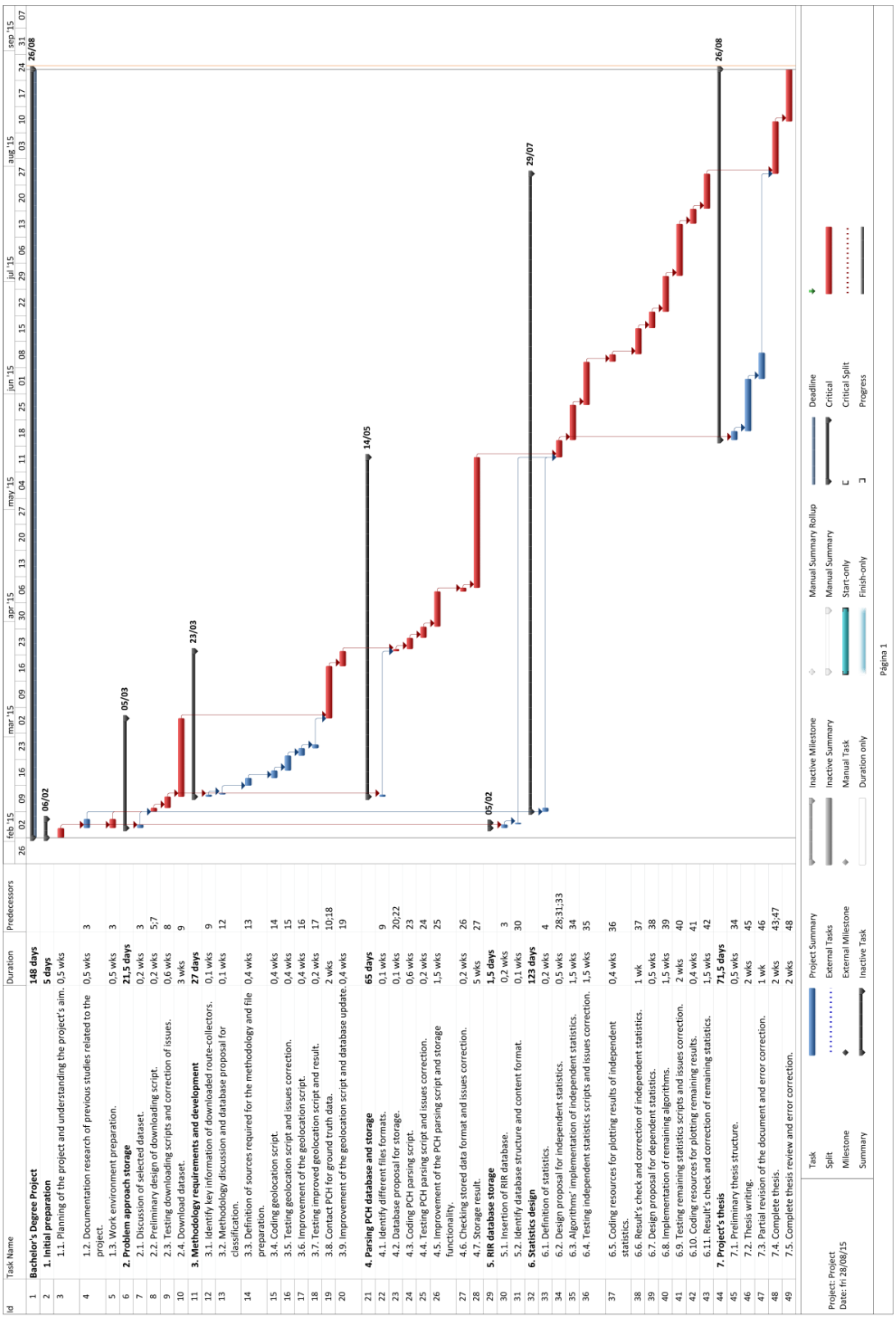
Milestones	Required tasks	Required time (in weeks)	Precedence
1 - Initial preparation			
1.1	Planning of the project and understanding the project's aim.	0.5	-
1.2	Documentation research of previous studies related to the project.	0.5	1.1
1.3	Work environment preparation.	0.5	1.1
2 - Problem approach storage			
2.1	Discussion of selected dataset.	0.2	1.1
2.2	Preliminary design of downloading script.	0.2	1.3, 2.1
2.3	Testing downloading scripts and correction of issues.	0.6	2.2
2.4	Download dataset.	3	2.3
3 - Methodology requirements and development			
3.1	Identify key information of downloaded route-collectors.	0.1	2.3
3.2	Methodology discussion and database proposal for classification.	0.1	3.1
3.3	Definition of sources required for the methodology and file preparation.	0.4	3.2
3.4	Coding geolocation script.	0.4	3.3
3.5	Testing geolocation script and issues correction.	0.4	3.4
3.6	Improvement of the geolocation script.	0.4	3.5
3.7	Testing improved geolocation script and result.	0.2	3.6
3.8	Contact PCH for ground truth data.	2	2.4, 3.7

3.9	Improvement of the geolocation script and database update.	0.4	3.8
4 - Parsing PCH database and storage			
4.1	Identify different files formats.	0.1	2.3
4.2	Database proposal for storage.	0.1	3.9, 4.1
4.3	Coding PCH parsing script.	0.6	4.2
4.4	Testing PCH parsing script and issues correction.	0.2	4.3
4.5	Improvement of the PCH parsing script and storage functionality.	1.5	4.4
4.6	Checking stored data format and issues correction.	0.2	4.5
4.7	Storage result.	5	4.6
5 - RIR database storage			
5.1	Insertion of RIR database.	0.2	1.1
5.2	Identify database structure and content format.	0.1	5.1
6 - Statistics design			
6.1	Definition of statistics.	0.2	1.2
6.2	Design proposal for independent statistics.	0.5	4.7, 5.2, 6.1
6.3	Algorithms' implementation of independent statistics.	1.5	6.2
6.4	Testing independent statistics scripts and issues correction.	1.5	6.3
6.5	Coding resources for plotting results of independent statistics.	0.4	6.4
6.6	Result's check and correction of independent statistics.	1	6.5
6.7	Design proposal for dependent statistics.	0.5	6.6
6.8	Implementation of remaining algorithms.	1.5	6.7
6.9	Testing remaining statistics scripts and issues correction.	2	6.8
6.10	Coding resources for plotting remaining results.	0.4	6.9
6.11	Result's check and correction of remaining statistics.	1.5	6.10
7 - Project's thesis			
7.1	Preliminary thesis structure.	0.5	6.2
7.2	Thesis writing.	2	7.1
7.3	Partial revision of the document and error correction.	1	7.2
7.4	Complete thesis.	2	6.11, 7.3
7.5	Complete thesis review and error correction.	2	7.4
	TOTAL DURATION	**28**	

Table 29: Milestones sequencing order of the project.

Note that the 3rd milestone does take into account plenty tests and corrections since it is the most important task in the project. If an error is not detected here, the dependent milestones would have to be done again from scratch. Therefore, the estimated time waiting for PCH ground truth data has to be considered when we contact with it. While we waited for it, we were able to set the basis of 4th milestone.

A.2. Gantt chart



Id	Task Name	Duration	Predecessors
1	**Bachelor's Degree Project**	**148 days**	
2	**1. Initial preparation**	**5 days**	
3	1.1. Planning of the project and understanding the project's aim.	0,5 wks	
4	1.2. Documentation research of previous studies related to the project.	0,5 wks	3
5	1.3. Work environment preparation.	0,5 wks	3
6	**2. Problem approach storage**	**21,5 days**	
7	2.1. Discussion of selected dataset.	0,2 wks	3
8	2.2. Preliminary design of downloading script.	0,2 wks	5;7
9	2.3. Testing downloading scripts and correction of issues.	0,6 wks	8
10	2.4. Download dataset.	3 wks	9
11	**3. Methodology requirements and development**	**27 days**	
12	3.1. Identify key information of downloaded route-collectors.	0,1 wks	9
13	3.2. Methodology discussion and database proposal for classification.	0,1 wks	12
14	3.3. Definition of sources required for the methodology and file preparation.	0,4 wks	13
15	3.4. Coding geolocation script.	0,4 wks	14
16	3.5. Testing geolocation script and issues correction.	0,4 wks	15
17	3.6. Improvement of the geolocation script.	0,4 wks	16
18	3.7. Testing improved geolocation script and result.	0,2 wks	17
19	3.8. Contact PCH for ground truth data.	2 wks	10;18
20	3.9. Improvement of the geolocation script and database update.	0,4 wks	19
21	**4. Parsing PCH database and storage**	**65 days**	
22	4.1. Identify different files formats.	0,1 wks	9
23	4.2. Database proposal for storage.	0,1 wks	20;22
24	4.3. Coding PCH parsing script.	0,6 wks	23
25	4.4. Testing PCH parsing script and issues correction.	0,2 wks	24
26	4.5. Improvement of the PCH parsing script and storage functionality.	1,5 wks	25
27	4.6. Checking stored data format and issues correction.	0,2 wks	26
28	4.7. Storage result.	5 wks	27
29	**5. RIR database storage**	**1,5 days**	
30	5.1. Insertion of RIR database.	0,2 wks	3
31	5.2. Identify database structure and content format.	0,1 wks	30
32	**6. Statistics design**	**123 days**	
33	6.1. Definition of statistics.	0,2 wks	4
34	6.2. Design proposal for independent statistics.	0,5 wks	28;31;33
35	6.3. Algorithms' implementation of independent statistics.	1,5 wks	34
36	6.4. Testing independent statistics scripts and issues correction.	1,5 wks	35
37	6.5. Coding resources for plotting results of independent statistics.	0,4 wks	36
38	6.6. Result's check and correction of independent statistics.	1 wk	37
39	6.7. Design proposal for dependent statistics.	0,5 wks	38
40	6.8. Implementation of remaining algorithms.	1,5 wks	39
41	6.9. Testing remaining statistics scripts and issues correction.	2 wks	40
42	6.10. Coding resources for plotting remaining results.	0,4 wks	41
43	6.11. Result's check and correction of remaining statistics.	1,5 wks	42
44	**7. Project's thesis**	**71,5 days**	
45	7.1. Preliminary thesis structure.	0,5 wks	34
46	7.2. Thesis writing.	2 wks	45
47	7.3. Partial revision of the document and error correction.	1 wk	46
48	7.4. Complete thesis.	2 wks	43;47
49	7.5. Complete thesis review and error correction.	2 wks	48

Figure 90: Gantt chart of the project.

ANNEX B: Budget

This annex is broken down in three sections. Firstly, we present a summarized list of the resources described in the 2nd Chapter. Secondly, we detail the human resources needed for developing this project. Thirdly, we show the budget of this project and its total cost.

B.1. Cost of the tools and physical resources

As described in section 2.2.3 and according to the Gantt chart in the previous annex (figure 90), this project requires a server and a laptop to be developed on time. The cost of these resources is:

- Laptop: 1,600€. A 21% Value-Added Tax (VAT) is included.
- Server: 4,500€. The hard disk expansion and a 21% VAT are included.

On the other hand, the cost of each software tool is listed:

- Python: Open Source.
- MySQL: Open Source.
- MATLAB: License required.
- OS Ubuntu (and incorporated programs): Open Source.
- GNU Screen: Open Source.
- Cron: Open Source.
- Google Charts: Open Source.
- Gedit: Open Source.
- Gimp: Open Source.
- OpenOffice: Open Source.

Among the software tools, just MATLAB requires a license. It can be acquired by the student due to the agreement in the University. MATLAB license's cost is 142.78€ (VAT included) and updates will not be received after a year if it is not renew.

Regarding the physical resources cost, we must take into account the lifetime of both the server and the laptop for computing the amortization. It will be computed linearly according to the next formula:

$$Amortization = \frac{Cost\ of\ the\ resource * Project's\ duration}{Lifetime\ period}$$

Since the lifetime of the server is around 10 years and the project's duration is 7 months, the amortization cost of the server rises up to 262.50€. Similarly, we obtain that the laptop's amortization is 155.56€ given a lifetime of 6 years However, we could have taken into account the cost of a rented server, which will decrease the actual server's cost.

B.2. Human resources

Speaking about the human resources for developing this project, it first requires a 4th year bachelor degree student in Telecommunication Technologies Engineering. The cost of the work realized by the student is approximately 40€/hour. Since the student has dedicated around 440h, the total student's cost adds up to 17,600€.

Moreover, since the project is not developed at the University, there are two required tutors instead of one. On the one hand, a tutor at the Institute must guide the student when a problem in a research study arises and supervise the work done. The estimated dedicated hours for these tasks are 80, given a cost of 75€/h, leads on to a 6,000€ cost. On the other hand, the academic tutor has dedicated approximately 40h for managing and reviewing the thesis. The cost of each dedicated hour of the academic tutor is around 65€. Hence, the total academic tutor's cost increases up to 2,600€.

B.3. Budget of the project

After considering the resources' cost, we must also include the indirect costs such as the Internet connection, power and water consumption, rent a premise, etc. The amount of this cost is computed as a 20% of the direct total cost. Thus, the budget is presented in table 30, with a total cost of 32,113.01€.

Concept	Quantity	Unitary cost	Project's duration	Lifetime	Total
Software licenses					
MATLAB license	1 unit	100€	7 months	-	142.78€
Physical resources					
Server	1 unit	4,500€	7 months	10 years	262.50€
Laptop	1 unit	1,600€	7 months	6 years	155.56€
Human resources					
Telecommunication Technologies Engineering student	440 h	40€/hour	7 months	-	17,600€
Institute's tutor	80h	75€/h	7 months	-	6,000€
Academic tutor	40h	65€/h	7 months	-	2,600€
Total direct cost					26,760.84€
Indirect cost (20%)					5,352,17€
TOTAL					32,113.01€

Table 30: Budget of the project.

ANNEX C: Regulatory environment

The realization of an engineering research project implies the technical development itself, and the adaption to the regulatory environment laid down both internally and at the sector level in any area. In this section the legal and technical environment are defined.

C.1. Legal environment

Since the research statistics are developed in an Advanced Research Institute promoted by the Madrid Regional Government, there are not many laws that could restrict the project development. However, all the work is subject to the Intellectual Property Law (LPI). Intellectual Property is a set of rights that belong to the authors and other title holders (artist, producers, etc.) about the works and services originated in its creation [71].

According to the No. 97.4 LPI article, there is a legal transfer of the exploitation rights to the Institute, which indicates this transfer in the cases when the author is a salaried employee of a computer program. Moreover, among the legal restrictions into LPI, it is also mandatory that the licenses for software tools such as MATLAB must be legalized and updated for obeying the actual LPI.

Besides, there is another regulation concerning the Information and Communication Technologies (ICT) sector [72] that must be obeyed, which is the Organic Law of Personal Data Protection (LOPD). LOPD's aim is to guarantee and protect the civil liberties and fundamental rights of individuals (specially their honor and both personal and familiar privacy [73]) regarding personal data processing. As all the resources to develop this project are publicly available on the Internet, as well as the PCH dataset, this regulation is obeyed.

Last but not least, we must guarantee Directive No. 95/46/CE of the European Parliament and the Council of 24 October 1995 [74], which is related to the European legal environment on the protection of individuals regarding the processing of personal data and the free movement of such data. This Directive attempts to guarantee individuals data protection and the citizen's privacy rights regarding data processing. Eventually, the legislation needs to be reformed, due to the quick technology growth and the evolution in terms of Big Data and Cloud Computing. Hence, the European Parliament has been working on a Proposal for General Data Protection [75] in order to update the terms to this era and regulate the European countries properly.

C.2. Technical environment

Concerning the technical section, there is an external rule which limits the execution of the project. While downloading the dataset, we took care of avoid aggressing the server and blocking

the public IP address of the Institute. Since the scripts are only used in the Institute, the remaining technical regulations are given by the internal project's realization rules.

Among these rules, we found that every database, file or result got has to be immediately stored in an external drive as a backup and follow a consistent structure. Every project script, statistic or graph must be reported at the very moment is completed in order to be reviewed. Once a statistic is reviewed, the script for obtaining it must be stored in a folder with the required additional files for running it. Afterwards, it should be briefly commented in a document for keeping track of the project's development.

With respect to consistency, all the scripts must be coded in Python and all the graphs which are related to dates or time must be developed in MATLAB (but in other projects there could be other languages). Moreover, the scripts should be compatible with the treatment of RouteViews data as well.

Note also that the restrictions that could have been occurred along the project's development are detailed in the corresponding section in order to be explained in their context.

Finally, the realization of the project provides scripts that could be easily reused for quickly downloading and parsing data, as well as an effective geolocation method of any dataset (which will have to be review if any route-collector is classified with two different CCs as in our case).

ANNEX D: Socio - economic environment

One of the key aspects in project's development (and especially in research works) is the environment where it is placed, since it will affect directly or indirectly to the development itself. In this section we discuss the significant aspects regarding the socio – economic environment and the benefits that the project supposes to it.

D.1. Social environment

From a social point of view, we first focus on the area the project is related to. Although the project is developed in an Advanced Research Institute promoted by the Madrid Regional Government, other Institutes and companies on the ICT sector are interested on these results. Therefore, the potential segment is ICT providers, decision makers and customers.

These results will surely be useful for the ISOC [76], since they could use them to take suitable decisions for the further investment in the African Internet. For instance, to elect a regional IXP based on the number of prefixes or ASes that are visible at that IXP, incentivize IXPs that are not growing according to PCH dataset. Next, the most interested stakeholders are ISPs and IXPs, since the work done shows the importance of peering with route-collectors. It allows performing studies on historical data leading to statistics such as the power of an IXP, showing the impact of ASes routing policies and incite new resources or infrastructure designs.

Therefore, this project significantly impacts the social environment and provides the basis for developing new studies on improving peering among African IXPs and decision making.

D.2. Economic environment

In this section, we focus on the project economic aspects regarding the Institute environment, since the statistics cannot be commercialized. So, the best way for understanding the economic benefits is by means of the business model of the Institute. Therefore, this section describes de Institute, its business model and the student's contribution to this model.

D.2.1. Institute description

IMDEA (Advanced Research Study Institutes from Madrid) Institutes were born due to the necessity of an institutional research environment. They were promoted by the Madrid Regional Government whose purposes were to realize excellence researches, improve the industrial sector competitiveness (by means of the technology transferred to it) and placed Madrid as one of the most innovative regions which generate knowledge for science, technology and research. For accomplishing this aim, seven independent institutes were created in different research areas (see figure 91).

Figure 91: IMDEA Institutes.

IMDEA Networks Institute [77] started at the end of 2007 in Leganés, Madrid. It was born as an independent non-profit research organism. It is oriented to gather an international team for developing fundamental cutting-edge science in every communication network area. Since it is composed by an international team, IMDEA Networks is an English-speaking Institute that is actually growing. The researchers contribute to determine network science's future and expect to reach a great impact on the future, providing innovative solutions to real problems. The researchers could be separated according the distinct lines of research: networked systems and algorithms, wireless networking, and energy-efficient networking.

Figure 92: IMDEA Networks research lines logos.

D.2.2. Business model

Since IMDEA Institutes are non-profit research organisms, they establish agreements with other companies for carrying out their lines of research. On the one hand, Madrid Community decides the relation with the Banks and other companies. For instance, IMDEA Networks Institute has collaboration agreements with the European Investment Bank and the Ministry of Economics and Competitiveness in order to defray the building, operation and cost equipment. On the other hand, around 60 – 80% of the funds are provided by Madrid Regional Government. The remaining funds are obtained by means of European projects.

There are some private projects (such as Cisco, NEC or Zed projects), which economic result varies on the realization of each project. In the medium term, the Institute pretends become less dependent on Madrid Community, obtaining external funds from other affiliated companies.

Among the actual Institute external projects [78], some of the most relevant related to Mobile Networks are MONROE, mmMAGIC, TEAM and TIGRE5-CM. Regarding Wireless Networks we can find CROWD or SEARCHLIGHT projects.

D.2.3. Business model contribution

This project belongs to networked systems and algorithms research line, since the statistics are obtained by means of different algorithms and scripts. The contribution to the business model is to provide statistics that are used for writing a paper and that are useful for the "Mapping and Measuring Western African Internet" [76], whose final result is adding value to the Institute.

As mentioned in the future work, there are additional possible studies that will complete the contribution to the Peering growth on the African Internet. In particular, the computed statistics allow ISOC to make suitable decisions for establishing regional IXP, improving and decreasing the cost of Internet connectivity. We also plan to compute different statistics based on data collected from *(i)* the set of IXPs in the country – National view, *(ii)* the set of IXPs in the region - Regional view *(iii),* the set of IXPs in the continent involved the PCH dataset – Global view. Consequently, our scripts which function offline are going to be improved, and integrated to the computational module of the web platform for "Inter-Domain IP routing economic analysis" developed by the research team in which we work at IMDEA Networks Institute.

In sum, the Institute purposes regarding realization of excellence researches and improvement of the industrial sector competitiveness are accomplished by means of all these statistics and the future ones.

ANNEX E: Summary

E.1. Introduction

In the related work, different studies have been assessing the Internet [1] [2] [3]. While some have tried to map the whole Internet, others have targeted some specific regions such as the US [4]. So far, the common specificity of these projects is to have under-involved African countries. A few studies have indeed been targeting the Interdomain topology on the African continent [6] [7].

More recently, some researchers [7] have highlighted the remaining lack of interconnectivity among local African ISPs, as well as the use of existing Internet eXchange Points (IXPs) and the appearance of some recently established ones. Our study complements these recent works as we analyze in detail the growth of African IXPs over the last decade. Hence, the motivation for this project was to provide various statistics based on historical routing data collected from Packet Clearing House (PCH) African IXPs that would be helpful for some Institutions to make suitable decisions and empowering the Internet in the AFRINIC region by a better understanding of the underlying relationships.

E.2. Problem approach

Firstly, we must justify the choice of PCH dataset. In the literature, many works have been relying on RouteViews raw data [5]. However, in RouteViews BGP feeds from a total of 11 IXPs are collected since 2004. Among those IXPs, only two are in Africa: KIXP – Kenya Internet Exchange (Kenya), JINX – Johannesburg Internet Exchange (South Africa). In contrast, our research has led us to find a more complete database regarding the African continent. Since 2003, PCH has been peering at 159 IXPs covering 52 countries in 5 regions (AFRINIC, APNIC, LACNIC, ARIN and RIPE NCC). As for the African continent, 8 countries and 11 IXPs including KIXP and JINX are also involved in this dataset (table 31).

CC	Country	Cities	IXPs
ZA	South Africa	Cape Town, Johannesburg, Durban	CINX, JINX, DINX,
KE	Kenya	Nairobi	KIXP
MZ	Mozambique	Maputo	MIX
EG	Egypt	Cairo	CAIX
MW	Malawi	Lilongwe	MIXP
SD	Sudan	Khartoum	SIxP
TN	Tunisia	Tunis	TunIXP
NG	Nigeria	Ibadan, Lagos	IBIXP, NIXP

Table 31: African IXPs involved in PCH dataset.

Moreover, PCH is planning to deploy new boxes at 3 IXPs: Gambia, Tanzania and Rwanda. It is clear that using PCH dataset gives more completeness to our study.

We must also introduce the basic terminology involved in this document (see table 32) and the resources required for developing this project (see table 33).

Concept	Description
Internet Service Provider (ISP)	It is a company which mainly provides Internet connection to their customers, including personal and business access to the Internet. The customers could be both, other ISPs or individuals [9] [10].
Peering	It is a voluntary interconnection of separate ISPs aiming to exchange traffic between the customers of each network [9].
Internet Protocol (IP)	Protocol by which data is sent between computers on the Internet. Each computer (also known as host) has at least one IP address that identifies them uniquely on the Internet [11].
Autonomous System (AS)	IP network or group of IP networks possessing its/their own independent route policy [9].
Border Gateway Protocol (BGP)	Protocol that allows the exchange of routing information between ASes [9].
Internet Exchange Point (IXP)	Physical network access point through which ISPs connect their networks and exchange traffic. This structure minimize the ISPs traffic which should be delivered to their transit provider, and also minimize the average per-bit delivery cost [12].
Route-collector	Also referred as Public Route Server, systems that are publicly accessible, often via Telnet. It is able to also run pings, traceroutes, and "show ip bgp" commands [13].
Country Code (CC)	Two-letter suffix developed to represent countries and dependent areas, for use in data processing and communications [14]. Example: US (United States)
Regional Internet Registry (RIR)	Organization that manages the allocation and registration of Internet number resources (IP addresses and AS) within a particular region of the world [15]. The RIR system evolved over time, eventually dividing the world into five RIRs.
AS-path	Sequence of followed ASes to reach a destination from a given IP address source [9].
Origin AS	It is the AS where the route is originated. It is normally placed on the right-most side of the AS path [20].
IPv4 address	Route IP address number for a destination [21].

Table 32: Terminology involved in the document.

Tool	Usage description
Ubuntu	It is a Free Software Operating System which includes most tools used in this project, mainly Python, MySQL, Screen and Cron.
Python	It is a programming language used for scripting and as a language to connect existing components together [22].
MySQL database	MySQL databases are multi-thread and can support many queries in parallel. Our algorithms use the parsed data which has been stored in databases.
GNU Screen	It is a tool to detach from and reattach to running terminals, so it is not needed to keep a running session with the running server. This tool was extremely useful when using a server in order to check the proper operation of a script running.
Cron	Cron is a job scheduler of processes in the background at regular intervals.

MATLAB	MATLAB [37] is a high-level computing language and an interactive environment for algorithm development, which allows to explore and visualize information. In the project, the software is used mostly for plotting 2D graphs.
Google Charts	It is Google's tool [44] for visualizing data. In the project, it was used to draw 3D Pie Charts.

Table 33: Tools involved in this project.

E.3. Data collection and classification methodology

We used Python to download and classify the dataset. When downloading PCH data, we realized that the webpage was being continuously modified (although not the data itself). So, we decided to download recursively the data per year in order to be able to check whether everything was downloaded for each year. We also decided to download in parallel the data which contributed to make our download script more efficient.

While downloading the dataset, we geolocated all the route-collectors given its name at PCH. This classification was difficult since a unique pattern has not been defined. The name of a PCH route-collector in general is under the formats: "router.<site code>.woodynet.net" or "route-collector.<site code>.pch.net". Note that <site code> is the nearest airport code in the first format, but it is not always the airport code in the second one. However, some <site codes> are a combination between city codes and IXP names.

It leads us to search for two letter CCs [53], city names [54], International Air Transport Association (IATA) codes [55] and IXP names [56] within the route-collectors names in order to geolocate them. Since this approach classified 121 route-collectors out of 159 (76.1%), we also searched for three letter CCs [57], but just 136 (85.53%) route-collectors were classified. Next, we pinged the route-collectors for extracting their IP addresses and we tried to geolocate them using the Open IP Map, Maxmind, Team Cymru, and Whois databases and a reverse DNS method. And we found that some route-collectors were not geolocated. Finally, we cross-checked the results with the ground truth deployment data obtained from PCH. In conclusion, the final methodology script geolocated correctly 98.11% of the route-collectors before cross-checking it with PCH data, which it is a great performance. We present an example of the geolocation methodology (see figure 93).

id	location	cc_code
1	route-collector.dac.pch.net	BD
2	optiglobe.woodynet.pch.net	US
3	route-collector.nlv.pch.net	UA
4	route-collector.nbo.pch.net	KE
5	route-collector.equinix-paris.pch.net	FR
6	200paul.woodynet.pch.net	US
7	route-collector.rno.pch.net	US
8	route-collector.cdg.pch.net	FR
9	route-collector.sin.pch.net	SG
10	nspixp2.woodynet.pch.net	JP

Figure 93: 10 first boxes geolocated in "locations" table.

E.4. Data parsing and storage

Then, we parsed all the data per year in parallel and stored it per region (given our geolocation results) on a server for further processing (cf. figure 94). Among the difficulties found, we noticed that the beginning of the valid data was different in some files, so specific parsing was needed for each data source before its storage. In addition, we found just one route-collector in 2003, and its file names and data format was completely different from all the other years. We finally decided to remove it and therefore, we will focus in this thesis on the remaining years (i.e. from 2005 to 2015 since no data was collected for 2004).

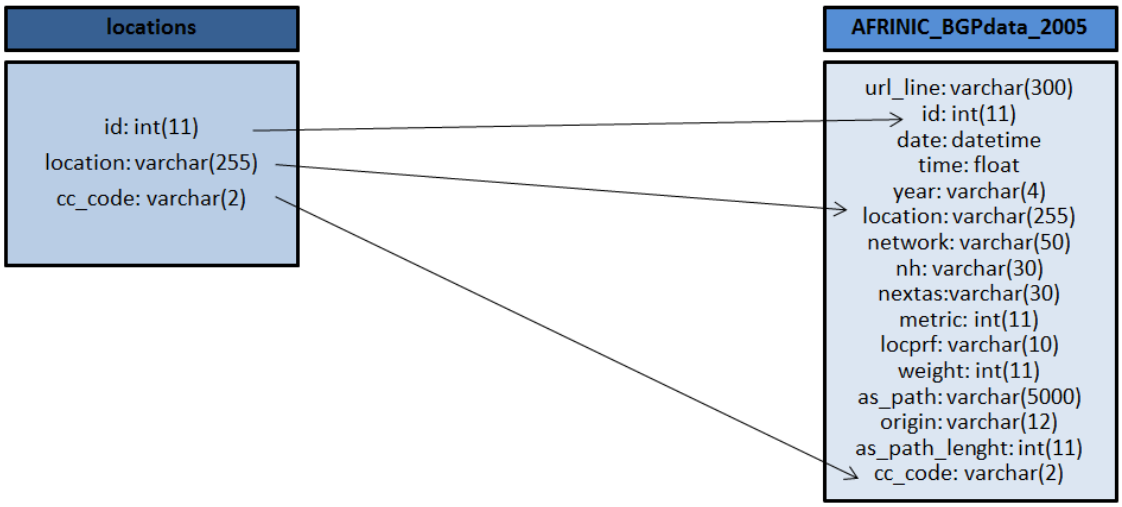


Figure 94: MySQL <Continent>_BGPdata_<year> table relation with "locations" table.

Since we wanted to analyze how African prefixes and Autonomous Systems (ASes) are seen from other countries, we also needed the information contained in the RIRs (Regional Internet Registry) databases. So we added the RIR database provided by the research team (see figure 95).

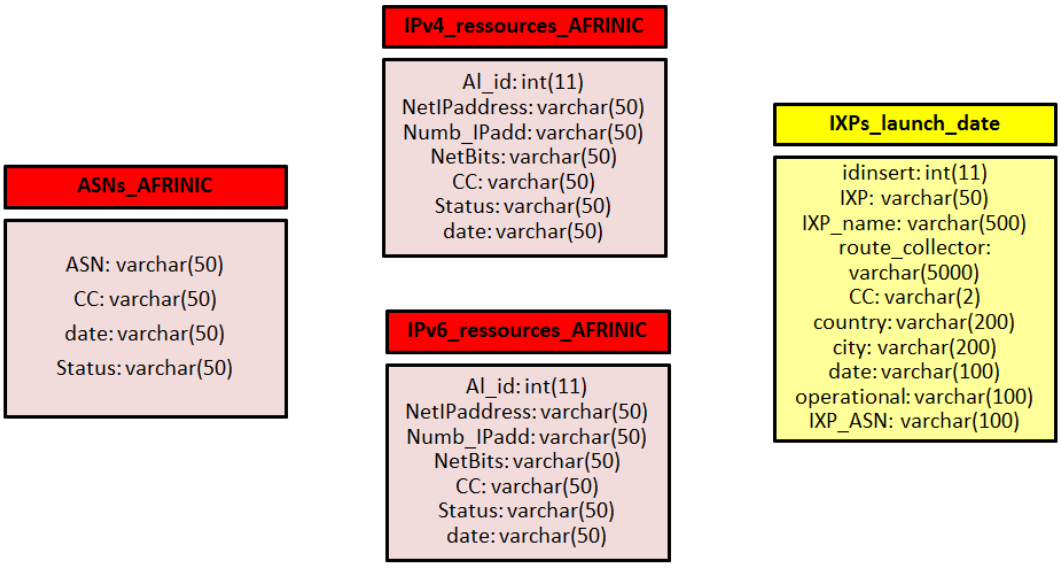


Figure 95: Complete relational RIRs database structure for African information.

E.5. Statistics (part I)

After all the required data was stored in the server, we performed the computations required for the distinct statistics in Python. We represented the results in MATLAB graphs and Pie Charts via HTML code. As some results did not need a graph, we used tables instead.

E.5.1. Time difference between prefix Allocation date and Appearance on the Internet

It is remarkable that in general Internet Assigned Numbers Authority (IANA) has granted 3,227 v4 prefixes to AFRINIC. Nevertheless, 3,067 are allocated or assigned. While treating the data, we found some prefixes that have been attributed twice by AFRINIC. Although AFRINIC allocates a prefix, it keeps ownership over that prefix. For instance, when it notices that an operator is not using an assigned prefix, AFRINIC may reallocate the prefix to another operator. It is also possible that if two companies merge into a new legal company, they will have to re-contract and re-allocate.

Taking this into consideration, we found about 268 prefixes reallocated. When they are, they are sometimes subnetted into prefixes of lower sizes (for instance 165.143.0.0/13 has been reallocated into a /16, etc.). So we took the first country where they have been appearing in our studies when considering the time gap between the first allocation date and the first appearance on the Internet. We computed this difference (see figure 96) and we found that it could be large since some prefixes had biased allocation dates in the AFRINIC delegated files (e.g. 00000000, years 1984, 1989, 1990, etc.), although AFRINIC was launched in 2004 [66]. About 7.4% prefixes are in such case.

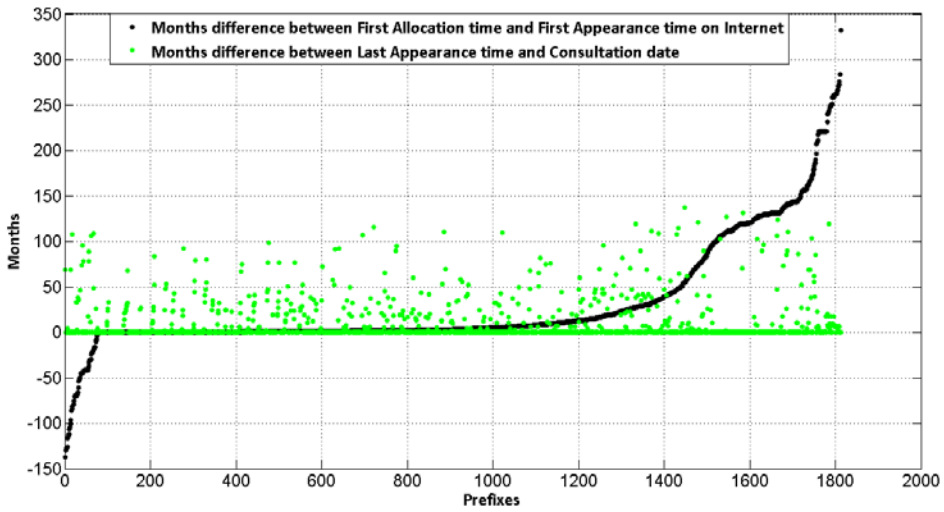


Figure 96: Months difference between the First Allocation time and the Consultation date (April 14, 2015) given RIPE stats for AFRINIC prefixes over time.

Most importantly, the graph also shows that 1,833 v4 prefixes out of the 3,067 have been allocated or assigned by AFRINIC to organizations in the region as of April 14th 2015. Among them, just 1,812 appear on Internet according to RIPE stats [65]. Hence, 98.85% of allocated v4 prefixes appear on the Internet. This graph also shows that only 25.28% of prefixes do not appear on the Internet on the consultation date. The remaining 1,354 appear at that date. Most importantly, 1,509 have appeared in 2015 as the last year of appearance, i.e. 83.28% of the prefixes under study appear in 2015.

We also provided the top four countries to which the prefixes first appearance is the first allocation date and also the prefixes whose first appearance is after a year of the first allocation date (cf. tables 34 and 35).

CC	Number and ratio of prefixes seen on Internet at the first allocation date
ZA	228 (12.58%)
KE	101 (5.57%)
EG	86 (4.74%)
NG	67 (3.70%)

Table 34: Top four countries with highest number of prefixes appearing at the first allocation date.

CC	Number and ratio of prefixes seen on the Internet a year after the first allocation date
ZA	75 (4.14%)
NG	37 (2.04%)
EG	26 (1.43%)
GH	21 (1.16%)

Table 35: Top four countries with highest number of prefixes appearing after a year from their first allocation date.

To sum up, 95.58% of v4 allocated prefixes appear on the Internet from their allocation date registered by AFRINIC. 87.12% have appeared lastly in 2015. Moreover, the prefixes that have appeared most are from South Africa, Nigeria and Egypt.

E.5.2. Time difference between prefix Allocation and Appearance in the data collected by PCH route-collectors deployed at any African IXP

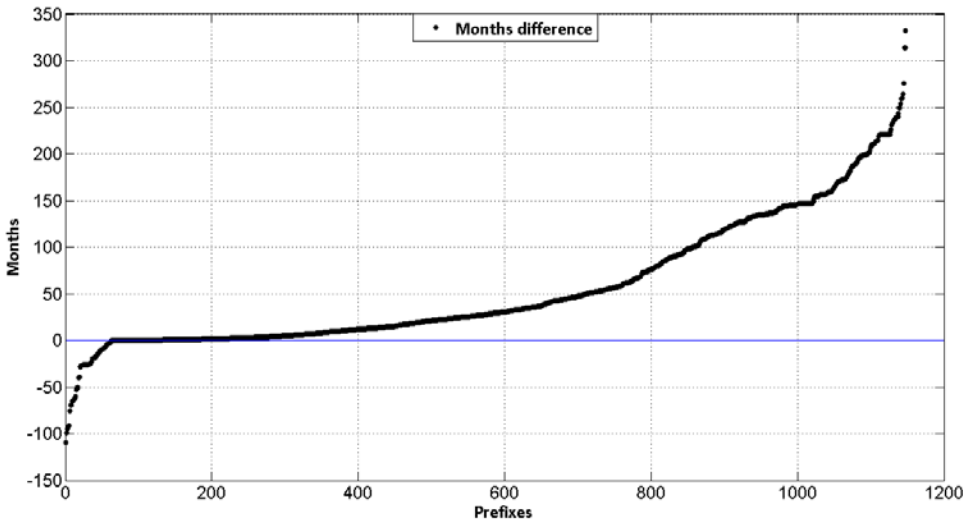


Figure 97: Months difference between the first allocation time and the first time appearance of prefixes at any IXP in the AFRINIC region.

At PCH boxes, we observe that 1,148 is the amount of IPv4 prefixes announced at any IXP in the AFRINIC region (see figure 97). Thus, the percentage of prefixes announced by ISPs that are peering with PCH boxes at African IXPs is 63.36%. In contrast with the information in RIPE stats, we have just a prefix whose months difference between the first allocation time and the first time appearance is null.

From the graph, it is easy to detect that more than a half of the prefixes are announced for the first time after a year of their allocation date. Just 28.92% prefixes shown are announced in the same year of the allocation date and 5.75% of the prefixes in the graph are announced before their allocation date. We also analyzed the time gap per IXP and we found that the IXPs mostly used for peering in PCH (JINX and CINX), have more reallocated prefixes than the rest of the IXPs. Moreover, the ones less used did not show any change in their graphs.

We also computed the number of distinct visible prefixes and distinct visible ASNs in the data collected by PCH at African IXPs (cf. tables 36 and 37). Despite there is not a huge difference between the origin ASNs visible at an IXP, we can see that KIXP has the biggest difference among all the IXPs. Besides, it is clear that all the ASes visible at the top three IXPs with less peering information are considering all the origin ASes found in each one of them.

CC	IXP	Number of visible prefixes
ZA	JINX	9690
EG	CAIX	6767
ZA	CINX	5412
KE	KIXP	4235
MZ	MIX	3323
ZA	DINX	1816
NG	NIXP	1140
SD	SIxP	496
MW	MIXP	107
TN	TunIXP	18

Table 36: Number of distinct visible prefixes per IXP.

CC	IXP	Number of distinct visible ASNs
ZA	JINX	541
KE	KIXP	540
ZA	CINX	500
ZA	DINX	177
MZ	MIX	132
EG	CAIX	74
NG	NIXP	69
MW	MIXP	14
SD	SIxP	9
TN	TunIXP	2

Table 37: Number of distinct visible ASNs per IXP.

E.6. Statistics (part II)

In this section we focus on the results that seem the most promising regarding the evolution over time. Among them, we studied the prefix and ASNs growth per year in the data collected by PCH at each African IXP. From those results, we observe that the newest IXPs (from 2013 onwards) are TunIXP, SIxP, NIXP, MIXP and DINX. However, taking into account their date of launch, the newest IXPs are TunIXP, SIxP, and DINX. Hence, they are still growing, whereas the IXPs who have been peering the earliest (JINX and KIXP) show a drop in evolution.

We developed an analysis of the unique number of prefixes and origin ASNs that appear at each African IXP in consecutive and non-consecutive years from 2005 to 2015 in PCH dataset too.

On the one hand, we saw that the top three IXPs that have more prefixes seen consecutively out of the total prefixes visible at the IXP are: CINX (72.38%), TunIXP (94.44%) and SIxP (70.56%). Moreover, we observed that the top three IXPs that have less prefixes seen in consecutive years out of the total prefixes visible at the IXP are: NIXP (since it is only peering in 2015), MIX (19.2%) and IBIXP (since we have not data of peering). JINX (58.44%), CAIX (56.51%) and MIXP (50.47%) were more or less equally distributed.

On the other hand, we saw that for the IXPs CAIX and TunIXP the ASNs announced are equally distributed between the consecutive and non-consecutive tables. However, the difference was noticeable for the consecutive percentages at SIxP (88.89%), MIXP (85.71%) and CINX (81%) with respect to the total number of origin ASNs visible at the IXP. From the point of view of the ASNs not seen in non-consecutive years, on the one hand, we can remark NIXP (100%), since we only have peering data in 2015. On the other hand, IXPs like KIXP (72.40%) and CAIX (54.05%) gave such results since they are growing in the last years. When we considered all the ASNs results we saw that they were pretty much equal to the origin ones.

We also computed the ratio of African ASNs assigned to the country visible at an IXP in PCH dataset (see table 38).

CC	IXP	African ASNs in the IXP's Country	ASNs in IXP	Intersection	Difference	Ratio of African ASNs assigned to the country visible at the IXP (%)
SD	SIxP	7	9	7	2	100
MW	MIXP	12	14	9	5	75
KE	KIXP	80	540	59	481	74
EG	CAIX	69	74	50	24	72
ZA	JINX	332	541	199	342	60
MZ	MIX	26	132	15	117	58
ZA	CINX	332	500	183	317	55
NG	NIXP	144	69	52	17	36
ZA	DINX	332	177	43	134	13
TN	TunIXP	13	2	0	2	0

Table 38: Ratio of African ASNs assigned to the country visible per IXP.

On the subject of the top three ratios of IXPs whose visible ASNs are also assigned to the country of that IXP, we remark that SIxP, MIXP and KIXP. This table also shows that some IXPs attract ISPs from other countries, creating hubs (e.g. JINX in ZA). In contrast, some IXPs such as NIXP, DINX and TunIXP, have a more national scope with varying degree of coverage (measured in percentage of national ASes connected to the IXP).

As a final statistic, we studied the ratio of ASNs by country assignment (local vs. external). We show the conclusions of the most specific IXP according to PCH dataset (see figure 98).

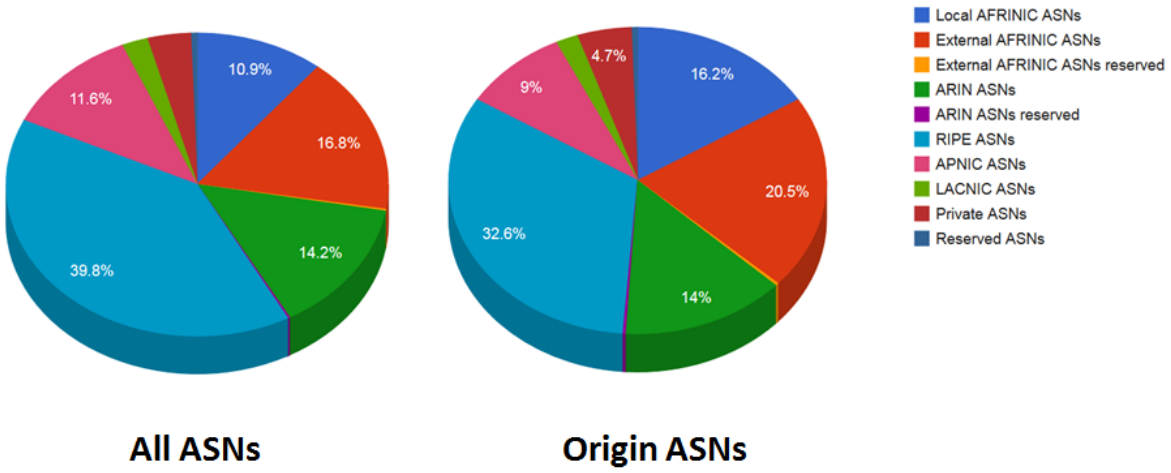


Figure 98: Ratio of ASNs by country assignment in routes collected by PCH boxes at KIXP (KE).

At KIXP, we have 16.2% of Origin ASNs local to the IXP, 20.5% external to the country, and 0.3% of them are reserved external in AFRINIC. ARIN, in this case, also has a percentage of 0.3% of Origin ASNs reserved and 14% that are already allocated or assigned to the region. Moreover, we found assigned 32.6% of them to RIPE, 9% to APNIC and 1.9% LACNIC. This is the only IXP that has both reserved ASNs for any other region that is not AFRINIC and assigned ASNs to LACNIC. Finally, we found that 4.7% of the Origin ASNs were private and 0.5% reserved. However, when we considered all the ASNs, the percentage of local ASNs to the IXP is 5.3% lower and the ratio of external ASNs in the AFRINIC region is 3.7% larger. RIPE is still the external region with the largest number of ASNs, followed by ARIN, APNIC and LACNIC, as before. The private ASNs ratio is reduced 1% and the LACNIC ASNs is 2.2%. Obviously, the reserved ASNs remain the same.

In summary, when studying the ratio of ASNs by country assignment (local vs. external), the top four IXPs ratio of ASNs local to the country are SIxP (77.8%), NIXP (73.2%), CAIX (67.6%) and MIXP (64.3%). The four highest IXPs percentage of external AFRINIC ASNs are DINX (65.4%), MIX (62.4%), TunIXP (50%) and CINX (42.1%).

Next, we focus on the top three IXPs ratio of ASNs that are not in the AFRINIC region since the fourth IXP does not present a significant ratio. The three highest IXPs percentage of ARIN ASNs are TunIXP (50%), SIxP (22.2%) and NIXP (18.3%) and MIXP (14.3%). Regarding the top three IXPs ratio of RIPE ASNs we find KIXP (39.8%), CAIX (8.1%) and JINX (7.9%). In addition, the top three IXPs ratio of APNIC ASNs are KIXP (11.6%), JINX (5.9%) and CINX (5.8%). Last but not least, we find that only one IXP has LACNIC ASNs which is KIXP (14.2%).

E.7. Conclusions

The main goal of this project was to provide diverse statistics based on historical routing data collected from African IXPs that would be helpful for some Institutions to make suitable decisions and empowering the Internet in the AFRINIC region by a better understanding of the underlying relationships.

It can be asserted that we have fulfilled this aim since we have found significant results and information. Our results show that 95.58% of the prefixes appear since their allocation date and 87.12% of them have appeared on 2015 as the year last of appearance. Moreover, the most frequent prefixes come from South Africa, Nigeria and Egypt. Also, the IXPs mostly used for peering (JINX and CINX according to our dataset) have more reallocated prefixes than the rest of the IXPs. In addition, the newest IXPs (from 2013 onwards) are TunIXP, SIxP, NIXP, MIXP and DINX. Nevertheless, taking into account their date of launch, the newest IXPs are TunIXP, SIxP, and DINX. Hence, they are still growing, whereas the IXPs who have been peering the earliest (JINX and KIXP) show a drop in the evolution. However, our dataset is biased since not every IXP in Africa is covered by PCH dataset. Although PCH has an open peering policy, not all ISPs at an IXP peer with PCH boxes. In addition, some route-collectors did not collect data every year up to 2015.

The most important difficulty found was to deal with so many different data sources when geolocating and storing the dataset, involving adaptations in format, interpretation and cleaning. Besides, we integrated many systems and languages to achieve the analysis, so our scripts were compatible among the different resources used.

Finally, our scripts are going to be integrated to the computational module of the web platform for “Inter-Domain IP routing economic analysis” developed by the research team in which the project was completed. Since the website will be publicly available in the near future, this web platform will be really helpful for taking suitable decisions aiming at empowering the Internet at any region. For instance, it would easily help ISPs to choose at which IXP to peer next, or be used by the Internet Society who would be able to determine as regional IXPs those at which we discovered most prefixes and origin ASes connected to and boost them.

References

[1] Z. M. Mao, J. Rexford, J. Wang, and R. Katz, “Towards an Accurate AS-level Traceroute tool,” in *Proceedings of ACM SIGCOMM*, pp. 365–378, 2003.

[2] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz, “Characterizing the Internet Hierarchy from Multiple Vantage Points,” in *Proceedings of IEEE INFOCOM*, p. 12, 2002.

[3] H. Haddadi, M. Rio, and A. Moore, “Network Topologies: Inference, Modeling, and Generation,” in *IEEE Communications Surveys and Tutorials*, pp. 48–69, IEEE, 2008.

[4] N. Spring, R. Mahajan, and D. Wetherall, “Measuring ISP topologies with Rocketfuel,” ACM SIGCOMM, August 2002.

[5] David Meyer. University of Oregon - RouteViews Archive Project. June 2004. URL: *http://routeviews.org*.

[6] G. Arpit, C. Matt, F. Nick, C. Marshini, C. Enrico, and K.-B. Ethan, “Peering at the Internet’s frontier: A first look at ISP interconnectivity in Africa,” *Passive and Active Measurement*. Springer International Publishing, 2014.

[7] R. Fanou, P. Francois, E. Aben, “On the Diversity of Interdomain Routing in Africa, In Proceedings of PAM”, March 2015.

[8] Packet Clearing House (PCH), Routes exchanged by peers at diverse IXPs in the world. URL consulted on February 2015: *https://www.pch.net/resources/data.php*. New URL since March, 2015: *https://www.pch.net/resources/Routing_Data/*.

[9] J.F. Kurose, K. W. Ross, “Computer networking: A Top-Down Approach”, in Pearson Education, Inc. 2010.

[10] Vangie Beal, ISP – Internet service provider. May 2015. URL: *http://www.webopedia.com/TERM/I/ISP.html*.

[11] M. Rouse, Internet Protocol definition. March 2008. URL: *http://searchunifiedcommunications.techtarget.com/definition/Internet-Protocol*.

[12] Techopedia, Internet Exchange Point (IXP). May 2015. URL: *http://www.techopedia.com/definition/27705/internet-exchange-point-ixp*.

[13] Network Engineering, Route servers and looking glasses – what are they? June 11, 2013. URL: *http://networkengineering.stackexchange.com/questions/1056/route-servers-and-looking-glasses-what-are-they*.

[14] International Organization for Standardization, “Country codes – ISO 3166”, 2015.URL: *http://www.iso.org/iso/home/standards/country_codes.htm*.

[15] Wikipedia, Registro Regional de Internet. May 25, 2015. URL: *https://es.wikipedia.org/wiki/Registro_Regional_de_Internet*.

[16] The Number Resource Organization, Getting Internet Number Resources. May 2015. URL: *https://www.nro.net/policies/getting-internet-number-resources*.

[17] Wikipedia, Overview of the Border Gateway Protocol. May 2015. URL: *http://en.wikipedia.org/wiki/Border_Gateway_Protocol*.

[18] J. C. Cardona, P. Francois, "Making BGP filtering a habit: Impact on policies", January 10, 2014. URL: *https://tools.ietf.org/html/draft-cardona-filtering-threats-02*.

[19] Cisco Systems, Configuring Basic BGP, May 2015. URL: *http://www.cisco.com/c/en/us/td/docs/switches/datacenter/sw/5_x/nx-os/unicast/configuration/guide/l3_cli_nxos/l3_bgp.html*.

[20] Informit, BGP Path Attributes, May 2015. URL: *https://www.informit.com/library/content.aspx?b=CCIE_Practical_Studies_II&seqNum=80.*

[21] E. Rosen, G. Nalawade, Border Gateway Protocol (BGP) Parameters, May 2015. URL: *http://www.iana.org/assignments/bgp-parameters/bgp-parameters.xhtml*.

[22] Python community, What is Python? Executive Summary, 2015. URL: *https://www.python.org/doc/essays/blurb/*.

[23] Python community, Comparing Python to Other Languages, 2015. URL: *https://www.python.org/doc/essays/comparisons/*.

[24] Tutorialspoint, Python – Command Line Arguments, 2013. URL: *http://www.tutorialspoint.com/python/pdf/python_command_line_arguments.pdf*.

[25] D. Hellmann, PyMOTW re - Regular Expressions, 2015. URL: *http://pymotw.com/2/re/*.

[26] Tutorialspoint, Python Basic Tutorial – Database Access, 2015. URL: *http://www.tutorialspoint.com/python/python_database_access.htm*.

[27] Tutorialspoint, Python Basic Tutorial – Date & Time, 2015. URL: *http://www.tutorialspoint.com/python/time_sleep.htm*.

[28] F. L. Drake et al, Python – Datetime Objects, 2015. URL: *https://docs.python.org/2/search.html?q=datetime&check_keywords=yes&area=default.*

[29] D. P. D. Moss. Python community, The Python Package Index - Package documentation: netaddr, 2015. URL: *https://pypi.python.org/pypi/netaddr*.

[30] Python Software Foundation. F. L. Drake et al, Python – Mathematical functions, 2015. URL: *https://docs.python.org/2/library/math.html*.

[31] V. Gite, Extracting tar.gz File, February 8, 2008. URL: *http://www.cyberciti.biz/faq/linux-unix-bsd-extract-targz-file/*.

[32] M. Rouse, Search SQL server – Database definition, 2015. URL: *http://searchsqlserver.techtarget.com/definition/database*.

[33] W3Schools, SQL Data Types for Various DBs, 2015. URL: *http://www.w3schools.com/sql/sql_datatypes.asp*.

[34] J. F. Courtney, D. B. Paradice, K. L. Brewer, J. C. Graham, "Database Systems for Management", 2015. URL: *https://textbookequity.org/oct/Textbooks/Courtney_DatabaseSystemsforManagement.pdf*.

[35] T. Arvin, Comparison of different SQL implementations, 2014. URL: http://troels.arvin.dk/db/rdbms/.

[36] MongoDB Community, 2015. URL: *https://www.mongodb.org/*.

[37] MathWorks, MATLAB, 2015. URL: *http://es.mathworks.com/products/matlab/*.

[38] R. Goering, "MATLAB edges closer to electronic design automation world", *EE Times*, April 04, 2004. URL: *http://www.eetimes.com/default.asp*.

[39] MathWorks, Pros and cons in working with MATLAB, 2002. URL: *http://www.mathworks.com/matlabcentral/newsreader/view_thread/35717*.

[40] Ubuntu, "The Ubuntu Story", 2015. URL: *http://www.ubuntu.com/about/*.

[41] M. Helmke. A. Graner, "The Official Ubuntu Book." 7th Edition, June 2012.

[42] Free Software Foundation, GNU Operating System – GNU Screen, 2010. URL: *http://www.gnu.org/software/screen/*.

[43] Ubuntu, Community Help Wiki: CrowHowto, 2015. URL: *https://help.ubuntu.com/community/CronHowto*.

[44] Google, Google Charts, May 26, 2015. URL: *https://google-developers.appspot.com/chart/interactive/docs/*.

[45] Ubuntu, Community Help Wiki: SSH Introduction, 2015. URL: *https://help.ubuntu.com/community/SSH*.

[46] GitHub, Community Help Wiki: SSH, 2015. URL: *https://help.github.com/articles/generating-ssh-keys/*.

[47] K. Finley, "What Exactly Is GitHub Anyway?" July 14, 2012 at TechCrunch. URL: *http://techcrunch.com/2012/07/14/what-exactly-is-github-anyway*/.

[48] W. E. Shotts, Jr. LinuxCommand – Permissions tutorial, 2015. URL: *http://linuxcommand.org/lts0070.php*.

[49] Cristalab, Tutoriales: Cómo utilizar el comando chown en Linux, April 4, 2008. URL: *http://www.cristalab.com/tutoriales/como-utilizar-el-comando-chown-en-linux-c54510l/*.

[50] A. Cabibbo, “Web Development for Bioinformatics. A Beginners Course for Biology and Bioinformatics Students”, chapter 2: The LINUX operating system – Setting up a Linux Web Server, 2013. URL: *http://www.cellbiol.com/bioinformatics_web_development/doku.php/chapter_2_-_the_linux_operating_system/the_linux_filesystem*.

[51] Clker, Clip_of_new_file clip art, June 2009. URL: *http://www.clker.com/clipart-3827.html*.

[52] G. Newell, Linux/Unix command: wget, 2015. URL: *http://linux.about.com/od/commands/l/blcmdl1_wget.htm*.

[53] Adrian World Design, Create your own countries list, 2015. URL: *http://www.countries-list.info/Download-List*.

[54] BookADayRoom, International airport code list, 2015. URL: *https://www.bookadayroom.com/airportcodes.html#international*.

[55] NationsOnline, IATA 3-letter codes, the location identifier code of airports and cities around the world, 2015. URL: *http://www.nationsonline.org/oneworld/IATA_Codes/airport_code_list.htm*.

[56] Wikipedia, List of Internet exchange points by size, 2015. URL: *https://en.wikipedia.org/wiki/List_of_Internet_exchange_points_by_size*.

[57] IATA, Airline and Airport Code Search, 2015. URL: *www.iata.org/publications/Pages/code-search.aspx*.

[58] E. Aben, Open IP Map data (found in GitHub), 2015. URL: *https://github.com/emileaben/django-openipmap*.

[59] MAXMIND, “GeoIP2: inteligencia de IP líder en la industria”, 2012. URL: *https://www.maxmind.com/es/home*.

[60] Team Cymru, IP to ASN mapping, 2015. URL: *http://www.team-cymru.org/IP-ASN-mapping.html*.

[61] WHOIS, 2015. URL: *https://www.whois.net*.

[62] RIPE NCC, List of Country Codes and RIRs, 2015. URL*:* *https://www.ripe.net/participate/member-support/info/list-of-members/list-of-country-codes-and-rirs*.

[63] Python Community, Python Documentation – Built-In Functions, 2015. URL: *https://docs.python.org/2/library/functions.html*

[64] Andrew, “Las black boxes reconocen música de los artistas en cualquier local”, May 2015. URL: *http://danzeria.com/2015/05/07/las-black-boxes-reconocen-musica-de-los-artistas-en-cualquier-local/*.

[65] RIPE NCC, RIPE stats database, 2015. URL: *ftp://ftp.ripe.net/pub/stats/*.

[66] AFRINIC, Press release: “AFRINIC Partners With ICANN on AFRICA DNS Business Exchange Programme”, 2015. URL: *http://www.afrinic.net/en/library/news/press*.

[67] Cisco Systems, “Explaining 4-Byte Autonomous System (AS) ASPLAIN and ASDOT Notation for Cisco IOS”, 2009. URL: *http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/border-gateway-protocol-bgp/white_paper_c11_516829.html*.

[68] PeeringDB, Exchange Points, 2015. URL: http://www.peeringdb.com.

[69] Wikipedia, Sistema Autónomo - ASN ranges, 2015. URL: *https://es.wikipedia.org/wiki/Sistema_aut%C3%B3nomo.*

[70] Google Developers, Gallery - Pie Chart, 2015. URL: *https://google-developers.appspot.com/chart/interactive/docs/gallery/piechart*.

[71] Grupo de Políticas Públicas y Regulación, “La gestión de derecho de propiedad intelectual en el entorno TIC.” Colegio Oficial de Ingenieros de Telecomunicación, 2014.

[72] Grupo de Políticas Públicas y Regulación, “Protección de datos y privacidad en el sector TIC.” Colegio Oficial de Ingenieros de Telecomunicación, 2014.

[73] Jefatura del Estado, “Vigente Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.” Boletín Oficial del Estado, 1999.

[74] Organización Mundial de la Propiedad Intelectual. Unión Europea, “Directiva 95/46/CE del Parlamento Europeo y del Consejo”, 1995. URL: http://www.wipo.int/wipolex/es/details.jsp?id=13580.

[75] Picón & Asociados Derecho y Nuevas Tecnologías S.L. Reglamento General de Protección de Datos, 2015. URL: http://rgpd.es/.

[76] R. de Miguel, “Mapping and Measuring West African Internet”, 2015. URL: *http://www.networks.imdea.org/research/projects/mapping-and-measuring-west-african-internet.*

[77] IMDEA Networks Institute, About Us – “IMDEA initiative”, 2015. URL: *http://www.networks.imdea.org/.*

[78] IMDEA Networks Institute, “Research projects”, 2015. URL: http://www.networks.imdea.org/es/investigacion/proyectos.

[79] W. Strunk JR. and E.B. White, “The elements of style”. Longman, 4th Edition, 1999.